Opinnäytetyö (AMK)

Tietojenkäsittely

Tietojärjestelmät

2012

Arto Kivinen

TUPAS-PANKKITUNNISTAUTUMINEN JA RADIOMIKROFONITIETOKANTA

TURUN AMMATTIKORKEAKOULU
TURKU UNIVERSITY OF APPLIED SCIENCES

OPINNÄYTETYÖ (AMK) | TIIVISTELMÄ

TURUN AMMATTIKORKEAKOULU

Tietojenkäsittely | Tietojärjestelmät

Toukokuu 2012 | 91 sivua

Ohjaaja: Anne Jumppanen

Arto Kivinen

TUPAS-PANKKITUNNISTAUTUMINEN JA RADIOMIKROFONITIETOKANTA

Opinnäytteen tuotos sisältyy WISE-projektiin (White Space Test Environment for Broadcast Frequencies). WISE-projekti on Tekes-rahoitteinen ja projekti jatkuu vuoden 2013 loppuun saakka. Opinnäytteen tavoitteena on toteuttaa Tupas-moduuli, Radiomikrofoni-moduuli ja radiomikrofonitietokanta. Moduulit toteutetaan Drupal-sisällönhallintajärjestelmään (CMS).

Tupas-palvelu on Suomen pankkien sopima menettelytapa todentaa käyttäjä. Todentamiseen käytetään verkkopankkitunnuksia. Jokainen tunnistautumiskerta on maksullista palveluntarjoajalle. Palveluntarjoaja voi pyytää käyttäjästä henkilötietoja pankilta. Henkilötietoja on käsiteltävä henkilötietolain mukaisesti. Pankki palauttaa vastaussanoman query string -merkkijonona. Jokainen tunnistautumistapahtuma on yksilöllinen ja vain palveluntarjoaja ja pankki tietävät salaiset avaimet tietojen varmentamiseksi.

Drupal-sisällönhallintajärjestelmä sisältää useita rajapintoja ja funktioita internet sovelluskehittäjille. Opinnäytteessä on tutkittu rajapintoja ja funktioita, jotka auttavat tietoturvan parantamiseen ja nopeuttavat sovellusten valmistamisessa. Drupalin tietokantarajapinta muodostaa tietoturvallisesti SQL-tietokantakyselyn.

Radiomikrofonitietokannan tarkoitus on sisältää rekisteröidyt radiomikrofonilaitteet, lupahakemukset, käytössä olevat taajuudet ja taajuuksien sijaintipaikka sekä aika. Tietokanta on normalisoitu, joka mahdollistaa uusien toimintojen lisäämisen tietokantaan tulevaisuudessa. Käyttäjien hallintaa varten on otettu käyttöön suurin osa Drupalin ominaisuuksista tietoturvaa lisäämään ja virhetilanteiden välttämiseksi.

ASIASANAT:

Drupal, Moduuli, CMS, Tupas, Radiomikrofoni, Tietokanta, Verkkopankkitunnus, Henkilölaki, Query string, Rajapinta, SQL, Normalisointi

BACHELOR´S THESIS | ABSTRACT

TURKU UNIVERSITY OF APPLIED SCIENCES

Degree Programme in Business Information Technology | Information Systems

May 2012 | 91 pages

Instructor: Anne Jumppanen

Arto Kivinen

TUPAS BANK AUTHENTICATION AND RADIO MICROPHONE DATABASE

This thesis is a part of the WISE-project (White Space Test Environment for Broadcast Frequencies). The WISE-project is Tekes funded and continues to the end of year 2013. The goal of this thesis is to design and builds a Tupas bank authentication module to ensure the user is a real and an intended person. The target is to design and build a radio microphone database and a radio microphone registration module to record registered radio microphones and in use of radio frequencies. Both modules are developed into a Drupal content manager system (CMS).

The Tupas bank authentication is a Finnish bank user authentication standard for internet service providers to ensure the user is real and to enable the service provider to request personal information from the user. When requesting personal information the service provider must act in accordance with the Finnish personal data act. The bank will collect a fee for every user authentication from the service provider. Every user authentication event is unique and only the service provider and the bank know the secret key. The bank returns requested information via query string.

The Drupal CMS includes many useful interfaces and functions for internet technology developers. In this thesis such interfaces and functions have been studied that will help to increase security and speed up developing modules. For example the Drupal Database API increases SQL database query security by reducing SQL injections.

The purpose of the radio microphone database is to save registered radio microphones, keep record of a radio microphone license, in using radio frequencies and locations of the frequency and time when frequency is in use. The radio microphone database is normalized to improve performance and to ease adding new features into the radio microphone database in future.

KEYWORDS:

Drupal, Module, CMS, Tupas, Radio microphone, Database, Online bank account, Finnish personal data act, Query string, Interface, SQL, Normalization

SISÄLTÖ

LIITTEET

KUVAT

KUVIOT

TAULUKOT

1 JOHDANTO

1.1. Tutkimusongelma

Vuonna 2011 keväällä Suomessa oli 59 000 radiomikrofonilaitetta ja vuonna 2010 kasvuvauhti oli 7 000 kappaletta vuodessa (Liikenne- ja viestintäministeriö 2011, 9). Suomen laki vaatii, että radiomikrofonilaitteet on rekisteröitävä. Nykyinen radiomikrofonin rekisteröinti tapahtuu tulostetun lomakkeen lähettämisen kirjeitse viestintävirastolle. Taajuusalueista 174 - 230 MHz ja 470 - 789 MHz ovat tarkoitettu radiomikrofonilaitteille. (Viestintävirasto, 2011b)

Tarkoituksena on sähköistää viestintäviraston radiomikrofonilaiterekisteröintilomake. Sähköistämisellä nopeutetaan ja helpotetaan radiomikrofonilaitteen rekisteröintiä. Seurauksena toivotaan radiomikrofonilaitteen omistajan rekisteröivän laitteensa todennäköisemmin. Viestintävirasto vaatii, että rekisteröijän henkilöllisyys varmennetaan. Henkilöllisyyden varmentamiseksi on päätetty käyttää suomalaisten pankkien Tupas-palvelua.

WISE-projektille on olennaista, että kaikki käytössä ja vapaana olevat taajuudet ovat tiedossa. Tietokantaan tallennetuilla taajuuksilla mahdollistetaan tv-lähetysten ja samaa taajuuksien käyttävien kognitiivisten laitteiden taajuuksien häiriötilanteen aiheuttavan yhteentörmäyksen. Yhteentörmäyksien estäminen auttaa häiriötilanteiden estämiseksi. Kognitiivinen laite osaa valita ja vaihtaa oikean taajuuden, mutta radiomikrofonilaite ei muuta taajuutta automaattisesti. Radiomikrofonilaitteiden taajuuden säätämisen pitää tapahtua käyttäjän toimesta. WISE-projektista ja radiomikrofoneista WISE-projekti kappaleessa.

Radiomikrofonisivuston tarkoitus on auttaa yksityistä käyttäjää löytämään nopeasti ja käyttäjäystävällisesti vapaana olevat taajuudet käyttäjän haluamasta kohteesta. Radiomikrofonisivuston kautta käyttäjä rekisteröi laitteensa ohjeistetusti. Samalla rekisteröityvät käytössä olevat taajuudet geolokaatiotietokantaan.

1.2. Tutkimussuunnitelma

Opinnäytteen toimeksiantajana on Fairspectrum Oy. Opinnäytteessä teoreettisena viitekehyksenä on internetsivuston käyttäjän vahvaa tunnistautumista varten Suomen pankkien Tupas-palvelun ominaisuudet ja vaatimukset. Tupas-palvelun ominaisuudet varmistavat tietoturvan käyttäjälle, palveluntarjoajalle ja pankille.

Drupal on valittu sisällönhallintajärjestelmäksi käyttäjien hallintaa varten. Sisällönhallintajärjestelmällä hallitaan internetsivustosta esimerkiksi käyttäjiä tai tekstiartikkeleita. Olennainen osuus opinnäytteessä on tutkia standardi Drupal-ytimen tietoturvaseikat ja ominaisuudet, jotka auttavat tai ovat hyödyllisiä käyttäjien hallintaan, Tupas-tunnistautumistapahtumalle, radiomikrofonitietokannalle ja radiomikrofonilaitteen rekisteröinnille.

Drupal tarjoaa useita ohjelmointirajapintoja ja funktioita. Drupalia kehitetään nopeammin kuin Drupalin dokumentaatiosivustoa. On tutkittava testaten ja havainnollistaen Drupalin tarjoamia funktioita sekä internetsivuston käyttöön vaikuttavia ominaisuuksia ja verrattava Drupalin internetsivustolla olevaan dokumentaatioon. Rajapintojen, ominaisuuksien ja toimintojen käyttöä pitää arvioida hyötyjä käyttää Drupalin tarjoamia, eikä ohjelmoida itse lähdekoodikirjastoja. Drupalin tarjoamia rajapintoja sekä lähdekoodikirjastoja otettuna käyttöön internetsivustossa, on internetsivuston kehittäjä sidoksissa Drupalin kehitys-syklien vaiheissa.

1.3. Tavoite

Radiomikrofonirekisteröintijärjestelmä tarjotaan Viestintävirastolle palveluna. Radiomikrofonitietokannan tarkoitus on tarjota WISE-järjestelmälle ajan tasalla oleva tietokanta, johon on tallennettu käytössä olevat taajuudet. Radiomikrofonisivusto on suunnattu yritys-, yhteisö-, yhdistys- ja yksityiskäyttöön.

Empiirisessä osuudessa on tavoitteena kehittää ja tuottaa 1) Tupas-moduuli, 2) radiomikrofonitietokanta ja 3) radiomikrofonirekisteröintimoduuli. Tupas-moduulin tavoite on, että on mahdollista ottaa moduulin käyttöön kuka tahansa standardi Drupal 7 -versiota käyttävä taho. Tupas-moduulin Tupas-tunnistautuminen toimii useaan suomalaiseen pankkiin ja tunnistautuminen tapahtuu tietoturvalliseksi säännösten mukaisesti.

Radiomikrofonitietokanta on suunniteltu palveluntarjoajan tarpeita vastaavaksi ja tietokanta on normalisoitu. Tarpeina radiomikrofonitietokannalla on säilyttää käyttäjän rekisteröidyt radiomikrofonilaitteet, vastaanottimet ja lupahakemukset. Tärkein tarve radiomikrofonitietokannalla on säilyttää käytössä olevat taajuudet ja taajuuksien käyttösijainnit sekä käyttöajat. Tarkoituksena on myös, että usealla yrityksen työntekijällä on mahdollista käyttää rekisteröityjä radiomikrofonilaitteita.

2 WISE-PROJEKTI

2.1. Määritelmä

WISE (White Space Test Environment for Broadcast Frequencies) on projekti, joka tutkii radio- ja televisiotaajuuksien tehokasta käyttöä kognitiiviradioiden avulla. Kognitiiviradio on laite, joka kykenee säätämään toimintataajuutensa ja toimintaparametrit ympäristön vaatimusten mukaan. Kognitiiviradio pystyy myös prosessoimaan ja lähettämään tietoa ympäristössä havaitsemistaan toisista radiolaitteista. (WISE 2011a)

Samankaltaisia projekteja on meneillään muun muassa Yhdysvalloissa ja Isossa-Britanniassa. WISE-projekti tavoittelee Suomelle korkeaa asemaa maailman radio- ja televisiotaajuuksien hyödyntämismenetelmissä. Projekti on Tekes-rahoitettu Trial-teknologiaohjelmassa vuoteen 2013 asti.

Viestintäviraston lakisääteinen vastuu on suojella tv-kanava- ja radiomikrofonitaajuuksien käyttäjien oikeuksia. Radiomikrofoni (myöhemmin langaton mikrofoni) on radiolaite, joka pystyy lähettämään äänisignaalia langattomasti. Langattomien mikrofonien taajuudet ovat siirtymässä samoille taajuusalueille televisiotaajuuksien kanssa, toisin sanoen radiomikrofonit voivat häiritä televisiokanavia, jos ne sattuvat samalle taajuudelle.

Langattomien mikrofonien rekisteröinti on todella vähäistä vanhentuneen rekisteröimistavan takia. Tämä tuo suuren riskin samojen taajuuksien käytölle useamman langattoman mikrofonikäyttäjän toimesta, sillä taajuuksien käyttöä ei voida valvoa, jos niiden käytöstä ei ilmoiteta viranomaisille.

Langattomien mikrofonien käyttämä taajuus voidaan valita manuaalisesti. Käyttäjät eivät yleensä tiedä taajuuksien signaaliliikenteestä, joten he voivat asettaa omat mikrofoninsa jo varatuille taajuuksille. Tämä johtaa useamman laitteen yhtäaikaisen taajuuden varaamiseen, jolloin voimakkaampi signaali saattaa yli-

ajaa heikomman ja jolloin toisen mikrofonin signaali ei pääse koskaan perille. Toisin sanoen toisesta mikrofonista ei kuulu ääntä.

2.2. Projektin tavoitteet

Yksi lyhyen tähtäimen tavoite WISE-projektissa on lisätä langattomien mikrofonien rekisteröinti-innokkuutta. Langattomien mikrofonien rekisteröinnin lisääntyminen vähentäisi taajuuksien samanaikaista käyttöä. Ratkaisuna rekisteröinnin vähyyteen WISE-projekti kehittää Internetissä toimivaa järjestelmää, joka mahdollistaa nopean ja tehokkaan sähköisen langattomien mikrofonien rekisteröinnin ja luvanhaun. (WISE 2011b).

Kognitiiviradiolaitteiden avulla taajuudet voidaan ottaa tehokkaaseen käyttöön, jotta samaa taajuutta voisi jakaa useammalle laitteelle eivätkä laitteet häiritsisi toisiaan, kun televisiotaajuuksien ja radiomikrofonien oikeudet ovat etusijalla. Tämä on toistaiseksi osoittautunut ongelmalliseksi, sillä langattomien mikrofonien päällä olo on vaikeasti ennustettavaa ja silloin kognitiiviset radiolaitteet eivät välttämättä havaitse niitä.

Tätä voidaan havainnollistaa kuvitteellisella tilanteella. Tietokanta jakaa tietyn taajuuden usealle käyttäjälle. Ongelma syntyy, jos kiinni ollut mikrofoni on aiemmin asetettu kyseiselle taajuudelle ja mikrofoni laitetaan päälle. Mikrofoni siirtyy automaattisesti aiemmin asetetulle taajuudelle, joka on nyt kognitiiviradiolaitteen jakamana käytössä. Jonkin laitteen tulisi siirtyä pois taajuudelta ja tämä on käytännössä vaikea toteuttaa.

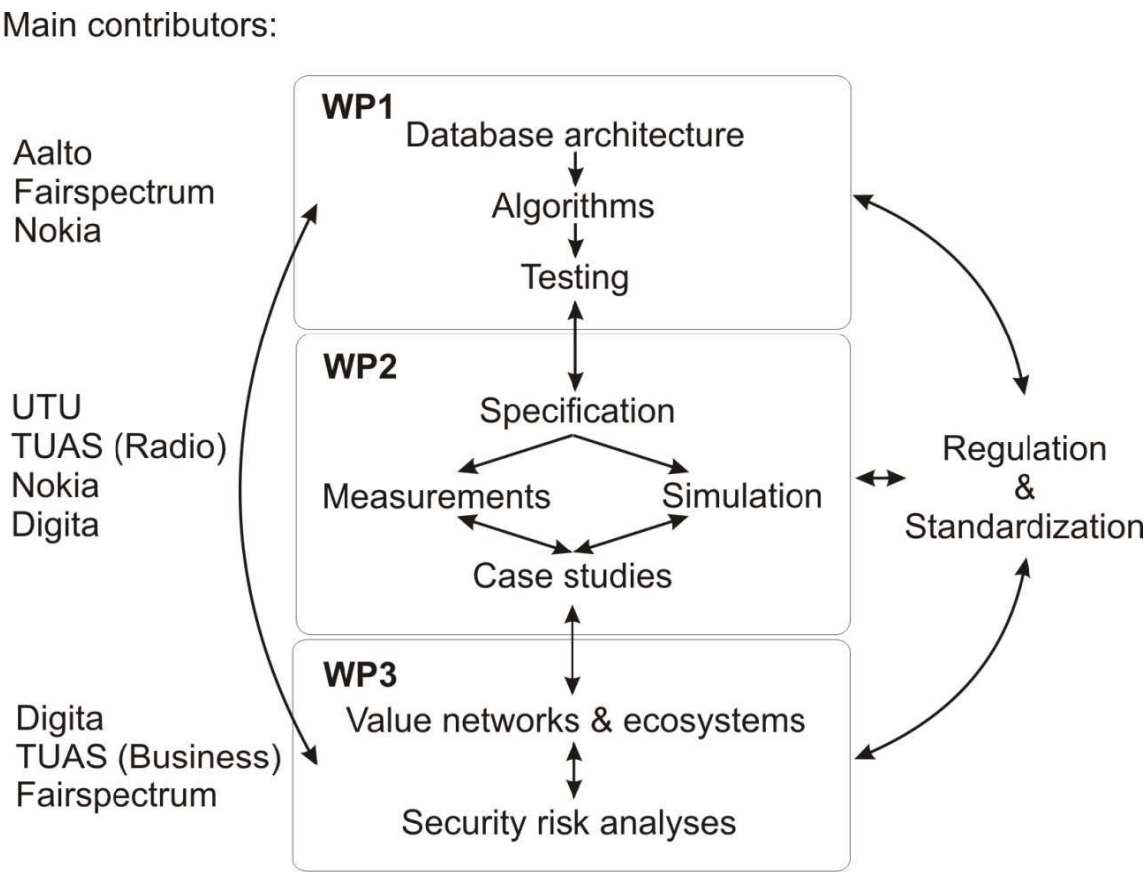
Kognitiiviradiolaitteiden toimintaperiaatteena on, että laite pyytäisi taajuuden lähetyslupaa WISE-projektin luomista ja ylläpitämistä tietokannoista. Tietokannoissa on kognitiiviradiolaitteen tarvitsema tieto televisio- ja radiolähetyksistä sekä vapaista taajuuksista. Mikäli lähetykselle ei ole tilaa taajuudella, lupa evätään.

Tietokantoja pyritään päivittämään koko ajan. Mikäli halutun taajuuden tietoa ei löydy tietokannasta niin kognitiiviradioiden tulisi kuunnella maantieteellistä ym-

päristöä, onko ympäristössä havaittavissa erilaisia signaaleja. Näin tietokantaa pystyttäisiin ylläpitämään reaaliaikaisesti. Tätä ei ole kuitenkaan vielä toteutettu.

2.3. Projektin jäsenet

WISE-projekti koostuu kolmesta työpaketista, jotka toteuttavat kahta osaprojektia: kognitiiviradioiden avulla ylläpidettävää testiverkkoa sekä taajuustiedot sisältävää tietokantaa (kuvio 1).



Kuvio 1. WISE-projektin rakenne (Paavola 2011).

WP1 eli ensimmäinen työpaketti koostuu tietokanta- ja algoritmikehityksestä sekä testauksesta. Aalto-yliopisto kehittää tietokanta-algoritmeja, joiden avulla taajuuksia voidaan jakaa kognitiiviradioille tehokkaasti. Helsinkiläinen yritys Fairspectrum tuottaa algoritmia käyttävän tietokantasovelluksen, joka toimii myös radiomikrofonirekisteröintijärjestelmän taustalla.

Toinen työpaketti sisältää vaatimusmäärittelyt, radiotaajuusmittaukset sekä projektissa tarvittavat simulaatiotutkimukset. Turun ammattikorkeakoulun insinööriopiskelijat ovat mukana projektissa tuottamassa radiotaajuusmittauksia. Turussa on testiverkko, jolla opiskelijat mittaavat kognitiivilaitteiden vaikutusta televisiolähetysten vastaanottoon.

Kolmanteen työpakettiin kuuluvat liiketalousasiat ja tietoturva. Turun ammattikorkeakoulun tietojenkäsittelyn koulutusohjelman opiskelijoita osallistuu taajuuslaskenta-algoritmin pohjalta luotua radiotaajuuksien geolokaatiotietokantaa käyttävien Internet-sivujen kehittämiseen. Tähän työpakettiin kuuluvat radiomikrofoneihin liittyvät työt ja niiden tietoturva.

3 TUPAS-PALVELU

3.1. Määritelmä

Tupas on Suomen pankkien sopima menettelytapa (FK 2011b). Tupas voidaan kirjoittaa isoilla kirjaimilla TUPAS tai pienillä kirjaimilla Tupas. Kaikki Suomen pankit eivät tarjoa Tupas-palvelun mahdollisuutta yksityis- tai yritysasiakkaille vuonna 2011. Jotta palveluntarjoaja saisi Tupas-palvelun käyttöön, on tehtävä jokaisen pankin kanssa palvelusopimus. (FK 2011a, 4)

3.2. Käyttötarkoitus

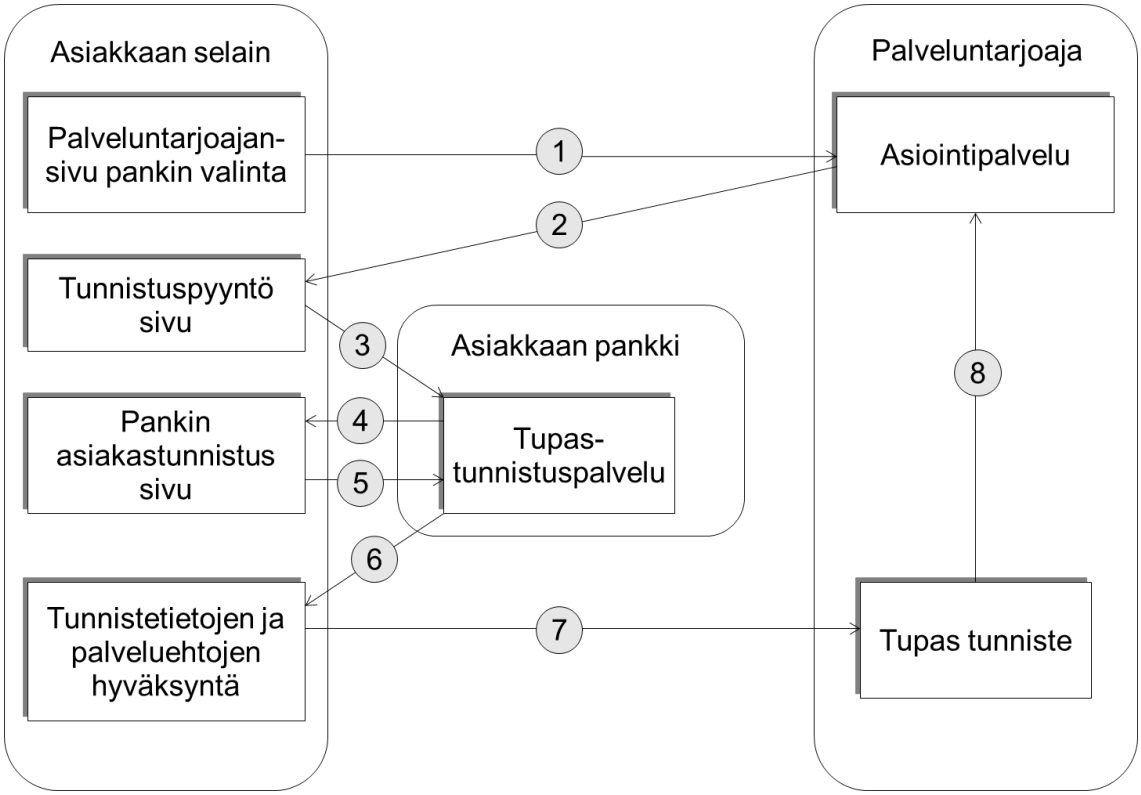
Asiakas voi internetpalvelussa tunnistautua verkkopankkitunnuksilla ja varmentaa henkilöllisyyden palveluntarjoajalle. Tupas-palvelun tunnistautumista voidaan kutsua vahvaksi tunnistautumiseksi. Tupas-palvelulla ei ole mahdollista tehdä maksutapahtumaa. Standardiltaan Tupas on samankaltainen kuin pankkien verkkopankkimaksupalvelun menettelytapa. Palveluntarjoaja voi hyödyntää Tupas-palvelua vähentäen käyttäjätietojen kirjoittamisen määrää internetpalveluun rekisteröidyttäessä, esim. henkilönimi ja henkilötunnus. Pankki palauttaa asiakkaasta aina henkilönimen. (Nordea Pankki Suomi Oyj 2011, 2)

Pankin ja palveluntarjoajan välinen sopimus määrää palautettavien tietojen sisällön Tupas-tunnistautuvasta henkilöstä tai yrityksestä. Arkaluonteisemmista asioista pitää tehdä selvitysilmoitus viranomaisille. Arkaluonteisiksi tiedoiksi katsotaan asiakkaan nimi tai henkilötunnus. Yrityksen tunnistautuessa voidaan pyytää pankilta y-tunnusta. (FK 2011a, 4)

3.3. Kuvaus järjestelmästä

Tupas-tunnistautumisen toimintaperiaatteena on, että internetsivulla on html form -lomake, jonka tietokenttien sisällöt lähetetään pankille. Pankin palvelusivuilla kirjaudutaan verkkopankkitunnuksilla pankin järjestelmään, minkä jälkeen pankki ohjaa käyttäjän takaisin palveluntarjoajan internetsivustolle. Käyttäjän ohjaava url-osoite sisältää pankilta tulevat tiedot. (FK 2011a, 5–7)

Tupas-palvelun kuvaus tapahtumajärjestyksien vaiheista on havainnollistettu kuviossa 2. Kuviossa 2 tapahtumien käyttäjänäkökulma on asiakkaan, koska internetpalvelussa Tupas-palvelun todennäköisin käyttäjä on palveluntarjoajan asiakas.



Kuvio 2. Tupas-palvelun järjestelmäkuvaus. (FK 2011a, 5)

Kuvion 2 vaihe 1. Asiakas siirtyy palveluntarjoajan internetsivulle, josta alkaa Tupas-tunnistautumistapahtuma. Kaikki tietoliikenne on salattu kuvion 2 vaiheesta 1 eteenpäin SSL-salaustekniikalla. (Nordea Pankki Suomi Oyj 2011, 6)

Kuvion 2 vaihe 2. Palveluntarjoaja lähettää tunnistautumiseen vaadittavat tiedot pankeista asiakkaan internetselaimeen. Jokaiselle pankkivaihtoehdolle palveluntarjoajan internetpalvelu tuottaa html form -lomakkeen. Jokainen html form -lomake on yksilöity MAC-turvatarkisteella ja aikaleimalla. Palveluntarjoajan on annettava asiakkaalle mahdollisuus nähdä internetselaimessa pankilta pyydettävät tiedot. Esimerkiksi asiakkaalle ilmoitetaan hänen henkilötunnuksensa pyytämistä pankilta. (FK 2011a, 5)

Kuvion 2 vaihe 3. Asiakkaan painettua html form -lomakkeen lähetä-painiketta internetselaimessa tiedot lähetetään asiakkaan pankille post-menetelmällä. Pankki tarkistaa MAC-turvatarkisteen lähetetyistä tiedoista tapahtuman varmistamiseksi. Virhetilanteessa pankki ohjaa asiakkaan takaisin palveluntarjoajan ilmoittamalle internetsivulle. (FK 2011a, 5)

Kuvion 2 vaihe 4. Pankki ohjaa asiakkaan pankin tunnistautumisen internetsivulle, jossa asiakas syöttää verkkopankkitunnukset html form -lomakkeeseen. Asiakkaalla on mahdollisuus peruuttaa tunnistautumistapahtuma, jolloin pankki ohjaa asiakkaan palveluntarjoajan ilmoittamalle internetsivulle. (FK 2011a, 6)

Kuvion 2 vaihe 5. Asiakas on lähettänyt kuvion 2 vaiheen 4 verkkopankkitunnukset pankkiin käsiteltäväksi. Virhetilanteessa pankki ohjaa asiakkaan pankin virheilmoitus internetsivulle, jossa asiakkaalla on mahdollisuus peruuttaa tunnistautumistapahtuma tai yrittää verkkopankkitunnuksien lähettämistä uudelleen. Asiakkaan peruuttaessa asiakas ohjataan palveluntarjoajan ilmoittamalle internetsivulle. (FK 2011a, 6)

Kuvion 2 vaihe 6. Pankki ohjaa asiakkaan internetsivulle, jossa näytetään asiakkaan tunnistustiedot ja pankin palveluehdot. Asiakkaan painaessa hyväksy-painiketta internetselaimessa asiakas hyväksyy palveluntarjoajalle lähetettävät tunnistustiedot sekä pankin palveluehdot. Asiakkaalla on mahdollisuus peruuttaa tunnistautumistapahtuma. (FK 2011a, 6)

Kuvion 2 vaihe 7. Asiakkaan tunnistustiedot lähetetään palveluntarjoajan internetsivulle. Palautettavat tiedot asiakkaasta on yksilöity MAC-turvatarkisteella ja aikaleimalla. Tunnistustiedot siirtyvät palveluntarjoajalle url-osoitteessa get-menetelmällä query string -merkkijonona. (FK 2011a, 6)

Kuvion 2 vaihe 8. Palveluntarjoaja tarkistaa asiakkaan pankin ohjaaman url-soitteen eheyden ja tapahtuman yksilöllisyyden. Url-osoitteen query string -merkkijonosta palveluntarjoaja tarkistaa MAC-turvatarkisteen ja aikaleiman, jonka palveluntarjoaja tallensi asiakkaan html form -lomakkeeseen kuvion 2 vaiheessa 3. (FK 2011a, 6)

Tupas-palvelun kuvion 2 vaiheiden 2 – 7 kaikkien osapuolten tietoliikenneyhteydet on suojattava SSL-salaustekniikalla. Jokaisen palveluntarjoajan ja asiakkaan väliset tietoliikenneyhteydet on salattava vähintään 128-bitin pituisella avaimella SSL-salaustekniikalla. Salausavaimen pituuden on mahdollista olla pidempi ja tämä määräytyy asiakkaan internetselaimen tukemista salaustekniikoista. Tietoliikenneyhteyden salauksella varmistetaan, että ulkopuoliset eivät näe siirrettävien tietojen sisältöä. (FK 2011a, 4)

3.4. Tunnistuspyynnön html form -lomake

Tupas-tunnistautumistapahtuman tiedot lähetetään pankille html form -lomakkeessa käyttäen post-menetelmää. Html form -lomake on samanlainen rakenteeltaan jokaisella pankilla. (FK 2011a, 7) Tupas-palvelu käyttää html form -lomakkeen elementeissä 8 bittistä ISO 8859-1 Latin1 -merkistöä (FK 2011a, 13).

Seuraava taulukko 1 sisältää html form -lomakkeen elementtien nimitykset ja selitteet elementtien sisältämistä arvoista. Tiedon nimi sarake ilmaisee html form -lomakkeen elementin nimen. Pituus sarake ilmaisee html form -lomakkeen elementin arvon pituuden merkkeinä.

Taulukko 1. Tunnistuspyynnön html form -lomake. (FK 2011a, 7)

Kenttä	Tiedon nimi	Pituus	Selite
1. Sanomatyyppi	A01Y_ACTION_ID	3 - 4	Aina vakio, 701
2. Versio	A01Y_VERS	4	Esim. ”0002”
3. Palveluntarjoaja	A01Y_RCVID	10 – 15	Palveluntarjoajan asiakas-tunnus
4. Palvelun kieli	A01Y_LANGCODE	2	ISO 639:n mukainen tunnus: FI = Suomi SV = Ruotsi EN = Englanti
5. Pyynnön yksi-löinti	A01Y_STAMP	20	Esim. yksilöinti aikaleimalla vvvvkkpphhmmssxxxxxx
6. Yksilöintitiedon tyyppi	A01Y_IDTYPE	2	Esim. ”02”
7. Paluuosoite	A01Y_RETLINK	-199	OK paluuosoite tunnisteelle
8. Peruuta-osoite	A01Y_CANLINK	-199	Paluuosoite peruutuksessa
9. Hylätty-osoite	A01Y_REJLINK	-199	Paluuosoite virhetilanteessa
10. Avainversio	A01Y_KEYVERS	4	Avaimen sukupolvitieto
11. Algoritmi	A01Y_ALG	2	01 = MD5 02 = SHA-1 03 = SHA-256
12. Tarkiste	A01Y_MAC	32 - 64	Pyynnön turvatarkiste

Taulukon 1 kenttä 1. Sanomatyyppi, joka määrittää pankille lähetetyn tiedon olevan tupas-tunnistuspyyntö. Sanomatyypin arvo on vakio Tupas-palvelussa. Sanomatyyppi tietokentän arvona on luku 701, joka on Tupas-palvelun tunnus-luku. (FK 2011a, 8)

Taulukon 1 kenttä 2. Tupas-tunnistautumistapahtuman tunnisteen versionume-ro. Versionumero on pankkikohtainen. Usealla pankilla on mahdollista olla sama versionumero käytössä. (FK 2011a, 8)

Taulukon 1 kenttä 3. Palveluntarjoajan tunnus, joka on pankkikohtainen. Pankki tunnistaa palveluntarjoajan asiakastunnuksen perusteella. Palveluntarjoajan tunnuksen perusteella pankki valitsee tietokannasta palveluntarjoajan salaisen

tunnusmerkkijonon, jonka avulla pankki tarkistaa MAC-turvatarkisteen (taulukko 1 kenttä 12). Ulkopuoliselle palveluntarjoajan tunnus ei ole hyödyllinen, koska ulkopuolisella ei ole tiedossa palveluntarjoajan ja pankin tietokannassa olevaa salaista tunnusta. (FK 2011a, 8)

Taulukon 1 kenttä 4. Tupas-palvelun kielikoodi määrittää pankille kielen, jolla asiakas haluaa asioida pankin internetsivuilla tunnistautumistapahtuman aikana. (FK 2011a, 8)

Taulukon 1 kenttä 5. Tunnistautumistapahtuman yksilöivä tunnus, jonka palveluntarjoaja määrittää jokaiselle html form -lomakkeelle internetsivua pyydettäessä. Pyynnön yksilöinti on mahdollista tehdä monella tavalla. Yksilöivänä tunnuksena voi olla viite, asiakasnumero, yhdistelmä päivämäärästä, yhdistelmä kellonajasta ja juoksevasta tunnuksesta. Suositeltava tunnus on aikaleima millisekunnin tarkkuudella. Pankki palauttaa palveluntarjoajalle palveluntarjoajan määrittämän yksilöivän tunnuksen tunnistuspyynnön vastaussanomassa. (FK 2011a, 8)

Taulukon 1 kenttä 6. Tunnistuspyynnön yksilöintitiedon tyyppikoodi, joka määrittää pankille palveluntarjoajan pyytämän henkilötiedon asiakkaasta. Yksilöintitiedon tyyppikoodi määräytyy pankin ja palveluntarjoajan välisestä sopimuksesta. (FK 2011a, 8)

Taulukko 2 ja taulukko 3 sisältävät yksilöintitiedon tyyppikoodin määritykset. X-kirjain on kymmenluku, joka määrittää pyydetyn yksilöintitiedon sisällön. Y-kirjain määrittää pyydetyn tunnisteen muodon. Yhdistelmäkoodia 43 ei ole käytössä. (FK 2011a, 14)

Taulukko 2. Yksilöintitunnuksen pyyntövaihtoehdot. (FK 2011a, 14)

Kympit X	Selite
0Y	Perustunnus
1Y	Henkilötunnus
2Y	Y-tunnus
3Y	Henkilötunnus tai y-tunnus
4Y	Henkilötunnus ja y-tunnus

Taulukko 3. Yksilöintitiedon palautettavat muodot. (FK 2011a, 14)

Ykköset Y	Selite
X1	Salattu tunnus
X2	Selväkielinen tunnus
X3	Typistetty tunnus

Taulukon 1 kenttä 7. Paluu url-osoite, joka määrittää palveluntarjoajan onnistuneen tupas-tunnistautumisen internetsivun. Pankki ohjaa asiakkaan onnistuneen verkkopankkitunnistautumisen jälkeen palveluntarjoajan onnistuneen tupas-tunnistautumisen internetsivulle. Url-osoitteen on oltava määritetty https-protokolla. (FK 2011a, 8)

Taulukon 1 kenttä 8. Peruuta url-osoite, joka määrittää palveluntarjoajan tunnistautumistapahtuman peruuttamisen internetsivun. Pankki ohjaa asiakkaan tupas-peruutus url-osoitteeseen, jos asiakas on peruuttanut tunnistautumistapahtuman pankin internetsivuilla. Url-osoitteen on oltava määritetty https-protokolla. (FK 2011a, 8)

Taulukon 1 kenttä 9. Hylätty url-osoite, joka määritetään pankin internetsivustolla tapahtunutta virhetilannetta varten. Asiakas ohjataan palveluntarjoajan tupas-virheinternetsivulle, jos pankin verkkopankkitunnistautumisessa tapahtuu virhe tai pankki hylkää asiakkaan verkkopankkitunnukset. Url-osoite on oltava määritetty https-protokolla. (FK 2011a, 8)

Taulukon 1 kenttä 10. Määrittää MAC-turvatarkisteen laskennassa käytettävän avaimen versionumeron. Pankki ilmoittaa palveluntarjoajalle käytössä olevan avaimen versionumeron. (FK 2011a, 8)

Taulukon 1 kenttä 11. Hash-algoritmin tunnus, joka on osa MAC-turvatarkisteen salausta. Hash-algoritmin MD5 tunnus on 01, joka tuottaa 32 merkin pituisen merkkijonon. Hash-algoritmin SHA-1 tunnus on 02, joka tuottaa 20 merkin pituisen merkkijonon. Tupas-palvelusta MD5- ja SHA-1 hash -algoritmit poistuivat käytöstä 31.12.2011. Hash-algoritmi SHA-256 tunnus on 03, joka tuottaa 64-merkin pituisen merkkijonon. (FK 2011a, 8)

Taulukon 1 kenttä 12. Määrittää tunnistepyynnön yksilöinnin ja eheyden pankin tarkistusta varten MAC-turvatarkistemerkkijonon. MAC-turvatarkiste muodostuu yhdistämällä taulukko 1 html form -lomakkeen elementtien 1‑11 arvot merkkijonoksi. Jokaisen elementin nimen ja arvon jälkeen lisätään & -merkki. Merkkijonon loppuun lisätään pankin ilmoittama salainen tarkisteavain, jonka jälkeen lisätään & -merkki. Merkkijonossa ei saa olla välilyöntejä. (FK 2011a, 8–9)

Merkkijonon määrittämisen jälkeen merkkijonosta lasketaan hash-merkkijono. Hash-algoritmin valinta perustuu A01Y_ALG-elementin tyyppikoodista. Hash-algoritmista palautuneesta heksadesimaalimerkkijonosta A – F -kirjaimet muutetaan isoiksi kirjaimiksi. (FK 2011a, 8–9)

Seuraavana on tunnistuspyynnön esimerkkimerkkijono, joka havainnollistaa taulukko 1 html form -lomakkeen elementtien arvot 1 – 11 merkkijonona, josta muodostuu MAC-turvatarkiste. Ensimmäinen esimerkkimerkkijono havainnollistaa merkkijonon rakenteen. Jälkimmäinen esimerkkimerkkijono havainnollistaa todellisen merkkijonon. MAC-turvatarkisteen yksilöivänä salaisena merkkijonona on Nordea pankin testitunnuksen LEHTI-merkkijono.

A01Y_RCVID& A01Y_LANGCODE&A01Y_STAMP&A01Y_IDTYPE&
A01Y_RETLINK&A01Y_CANLINK& A01Y_REJLINK&A01Y_KEYVERS&
A01Y_ALG&tarkisteavain&

701&0002&87654321&FI&20120321144413&02&http://www.esimerkki.fi/paluu.php&http://www.esimerkki.fi/peruutus.php&http://www.esimerkki.fi/virhe.php&0001&03&LEHTI&

Seuraavana on php-koodiesimerkki, joka muodostaa MAC-turvatarkisteen. Implode-funktio yhdistää taulukko 1 html form -lomakkeen elementtien 1 – 11 arvot ja pankin salaisen tarkisteavaimen merkkijonoksi. Hash-funktio muodostaa MAC-turvatarkisteen. Strtoupper-funktio muuttaa A – F -merkit isoiksi kirjaimiksi.

```
$macMerkkijono = implode('&', $lomake) .'&'. 'LEHTI' .'&';
$macMerkkijono = hash("sha256", $macMerkkijono, false)
$lomake['MAC'] = strtoupper( $macMerkkijono );
```

Html form -lomakkeen elementtien pitää olla piilotettuja elementtejä internetselaimessa. Mikään html form -lomakkeen elementeistä ei sisällä salaista tietoa palveluntarjoajasta tai asiakkaasta. (FK 2011a, 7)

Seuraavana on esimerkki, joka havainnollistaa html form -lomakkeen koodin rakenteen. Html form -lomakkeen elementtien nimet kirjoitetaan isoilla kirjaimilla.

```
<FORM METHOD="POST" ACTION="Pankin Tupas-palvelun url-osoite">
  <INPUT NAME="A01Y_ACTION_ID" TYPE="hidden" VALUE="701" />
  <INPUT NAME="A01Y_VERS" TYPE="hidden" VALUE="..." />
  <INPUT NAME="A01Y_RCVID" TYPE="hidden" VALUE="..." />
  <INPUT NAME="A01Y_LANGCODE" TYPE="hidden" VALUE="..." />
  <INPUT NAME="A01Y_STAMP" TYPE="hidden" VALUE="..." />
  <INPUT NAME="A01Y_IDTYPE" TYPE="hidden" VALUE="..." />
  <INPUT NAME="A01Y_RETLINK" TYPE="hidden" VALUE="..." />
  <INPUT NAME="A01Y_CANLINK" TYPE="hidden" VALUE="..." />
  <INPUT NAME="A01Y_REJLINK" TYPE="hidden" VALUE="..." />
  <INPUT NAME="A01Y_KEYVERS" TYPE="hidden" VALUE="..." />
  <INPUT NAME="A01Y_ALG" TYPE="hidden" VALUE="..." />
  <INPUT NAME="A01Y_MAC" TYPE="hidden" VALUE="..." />
  <INPUT TYPE="submit" VALUE="Lähetä" />
</FORM>
```

3.5. Tunnistuspyynnön vastaussanoma

Pankki lähettää asiakkaan tunnistetiedot palveluntarjoajalle url-osoitteen query string -merkkijonossa asiakkaan internetselaimen kautta get-menetelmällä. Palveluntarjoajan on huolehdittava, että asiakkaan ohjaava url-osoite, joka sisältää pankilta lähetetyt asiakkaan tunnistetiedot, ei jää asiakkaan selaimeen näkyviin internetselaimen url-osoitekenttään.

Taulukko 4 sisältää vastaussanoman query string -merkkijonon sisältämät elementit. Taulukon 4 sarake 1 ilmaisee tiedon pakollisuutta vastaussanomassa. P-kirjain on pakollinen tieto vastaussanomassa ja V-kirjain tarkoittaa vain palveluntarjoajan pankilta pyydettäessä. Taulukon 4 tiedon nimi sarake ilmaisee html form -lomakkeen elementin nimen.

Taulukko 4. Tunnistuspyynnön vastaussanoma. (FK 2011a, 9)

Kenttä	**Tiedon nimi**	**Pituus**	**1)**	**Selite**
1. Versio	B02K_VERS	4	P	Esim. ”0002”
2. Tunnisteen yksilöinti	B02K_TIMESTMP	23	P	NNNvvvvkkpphhmmssxxxxxx
3. Tunnisteen numero	B02K_IDNBR	10	P	Pankin tunnisteelle antama numero
4. Pyynnön yksilöinti	B02K_STAMP	20	P	Tunnistepyynnön taulukko 1 kenttä 7 A01Y_STAMP
5. Asiakas	B02K_CUSTNAME	-40	P	Tunnistetun henkilön tai yrityksen nimi.
6. Avainversio	B02K_KEYVERS	4	P	Avaimen sukupolvitieto
7. Algoritmi	B02K_ALG	2	P	01 = MD5 02 = SHA-1 03 = SHA-256
8. Yksilöintitieto	B02K_CUSTID	-64	P	Perustuu Taulukko 1 kenttä 6 A01Y_IDTYPE Esim. 583219-8851

(jatkuu)

Taulukko 4 (jatkuu)

Kenttä	Tiedon nimi	Pituus	1)	Selite
9. Yksilöintitiedon tyyppi	B02K_CUSTTYPE	2	P	Esim. ”05”
10. Käyttäjän tunnus	B02K_USERID	-64	V	Yrityskäyttäjän henkilötunnus tai salattu tunnus.
11. Käyttäjän nimi	B02K_USERNAME	-40	V	Yrityskäyttäjän nimi
12. Tarkiste	B02K_MAC	32 - 64	P	Vastauksen turvatarkiste

Taulukon 4 kenttä 1. Tupas-tunnisteen versionumero, joka on pankkikohtainen. Usealla pankilla on mahdollista olla samanlainen versionumero käytössä. (FK 2011a, 10)

Taulukon 4 kenttä 2. Vastaussanoman yksilöintitieto, joka sisältää pankin tupas-tunnusnumeron ja pankin järjestelmään tallennetun tapahtuman kirjauksen aikaleiman. NNN-merkit ilmaisevat pankin tupas-numerotunnusta. (FK 2011a, 10) Seuraavana on taulukko 5, joka sisältää Suomen pankkien NNN-merkkien tupas-numerotunnukset.

Taulukko 5. Pankkien tunnusnumerot. (FK 2011a, 10)

NNN	Pankin Nimi
200	Nordea Pankki Suomi
310	Handelsbanken
360	Tapiola Pankki
390	S-Pankki
400	Aktia, Säästöpankki ja Paikallisosuuspankit
500	Osuuspankkiryhmä
600	Ålandsbanken
800	Sampo Pankki

Taulukon 4 kenttä 3. Tunnisteen numerosarja on tunnistustapahtumalle annettu yksilöity tunnus. Tunnusnumero on rekisteröity pankin sisäiseen järjestelmään. (FK 2011a, 10)

Taulukon 4 kenttä 4. Tunnistuspyynnön yksilöintitunnus, jonka palveluntarjoaja on määrittänyt taulukon 1 kenttään 7 A01Y_STAMP-elementtiin. Pankki lähettää palveluntarjoajalta saadun yksilöintitietotyypin muuttumattomana tunnistuspyynnön vastaussanomassa. (FK 2011a, 10)

Taulukon 4 kenttä 5. Pankin tietokannassa rekisteröity tunnistettavan asiakkaan nimi tai yrityksen nimi. Pankki lähettää vastaussanoman tiedoissa asiakkaan henkilönimen tai yrityksen nimen. Yrityksen tupas-tunnistautuessa pankki ei lähetä palveluntarjoajalle tunnistautujan henkilönimeä. (FK 2011a, 10)

Taulukon 4 kenttä 6. Vastaussanoman avainversio määrittää MAC-turvatarkisteen laskennassa käytetyn avaimen versionumeron. Avaimen versionumeroa kutsutaan MAC-tarkisteavaimen sukupolvitiedoksi. (FK 2011a, 10)

Taulukon 4 kenttä 7. Hash-algoritmin tunnusnumero, jota pankki on käyttänyt MAC-turvatarkisteen salaamisessa. Hash-algoritmin tunnusnumero määrittää palveluntarjoajan valitsemaan oikean hash-algoritmin MAC-turvatarkisteen varmistamiseksi. MAC-turvatarkisteella palveluntarjoaja varmistaa query string -merkkijonon eheyden ja vastaussanoman lähettäjän alkuperän. (FK 2011a, 10)

Taulukon 4 kenttä 8. Vastaussanoman yksilöintitieto sisältää asiakkaan tunnuksen tai yrityksen y-tunnuksen. Yksilöintitiedon sisällön valinta perustuu tunnistuspyynnön taulukon 1 kentän 6 A01_IDTYPE-tunnusnumeroon. Asiakkaan yksilöintitiedon on mahdollista olla selväkielinen tai salattu. Yksilöintitiedon pitkä tunnus on esim. 583219-8851 tai lyhennetty tunnus on esim. 8851. Yrityksen y-tunnus välitetään muodossa xxxxxxx-x väliviivan kanssa esim. 1234567-8. (Nordea Pankki Suomi Oyj 2011, 10)

Taulukon 4 kenttä 9. Vastaussanoman yksilöintitiedon koodinumero, jonka tieto perustuu taulukon 1 kentän 6 A01Y_IDTYPE-elementin kooditunnukseen.

Pankki määrittää koodinumeron pankin järjestelmästä löytyneistä tiedoista ja tunnistuspyynnön A01Y_IDTYPE-elementin koodista. (FK 2011a, 15)

Seuraavana on taulukko 6 ja taulukko 7, jotka havainnollistavat yksilöintitiedon koodinumeroiden merkitykset. Taulukon 6 sarake 0Y ilmaisee, että asiakkaasta on löytynyt pyydetyt tiedot pankin järjestelmästä. Taulukon 7 sarake 1Y, joka on koodinumeron kymmenluku ja ilmaisee tunnistuspyynnössä yksilöintitiedon ja henkilötietojen tai yritystietojen kaikki tai osa pyydetyistä tiedoista ei löytynyt pankin tietokannasta. (FK 2011a, 15)

Taulukko 6. Yksilöintitiedon koodimääritelmät 0Y. (FK 2011a, 15–16)

0Y	**Selite – Pyydetyt tiedot on löydetty**
00	Tunnus ei ole tiedossa. 00-koodinumeroa käytetään, jos mitään tunnistetta ei löytynyt pankin tietokannasta.
01	Selväkielinen henkilötunnus. Pyydetty selväkielistä tunnusta ja palautetaan vain henkilötunnus. Vastaussanoman kentässä 5 on henkilön nimi ja kentässä 8 on selväkielinen henkilötunnus.
02	Selväkielinen henkilötunnuksen tarkenne. Pyydetty typistettyä tunnusta ja palautetaan vain henkilötunnuksen tarkenne. Vastaussanoman kentässä 5 on henkilön nimi ja kentässä 8 on selväkielisen henkilötunnuksen loppuosa.
03	Selväkielinen Y-tunnus. Pyydetty selväkielistä tunnusta ja palautetaan vain y-tunnus. Vastaussanoman kentässä 5 on yrityksen nimi ja kentässä 8 on selväkielinen y-tunnus.
04	Selväkielinen sähköinen asiointitunnus. Pyydetty selväkielistä tunnusta ja palautetaan vain sähköinen asiointitunnus. Vastaussanoman kentässä 5 on yrityksen nimi ja kentässä 8 on selväkielinen asiointitunnus.
05	Salattu henkilötunnus. Pyydetty salattua tunnusta ja palautetaan henkilötunnus. Vastaussanoman kentässä 5 on henkilön nimi ja kentässä 8 on salattu henkilötunnus.

(jatkuu)

Taulukko 6 (jatkuu).

0Y	Selite – Pyydetyt tiedot on löydetty
06	Salattu Y-tunnus. Pyydetty salattua tunnusta ja palautetaan vain y-tunnus. Vastaussanoman kentässä 5 on yrityksen nimi ja kentässä 8 on salattu y-tunnus.
07	Salattu sähköinen asiointitunnus. Pyydetty salattua tunnusta ja palautetaan vain sähköinen asiointitunnus. Vastaussanoman kentässä 5 on asiakkaan nimi ja kentässä 8 on salattu sähköinen asioinnintunnus. Sampo-pankilla ja Nordea-pankilla ei ole 07 kohtaa käytössä.
08	Selväkielinen Y-tunnus ja selväkielinen yrityskäyttäjän henkilötunnus, tai pankin ja palveluntuottajan keskenään sopima tunnus. Pyydetty selväkielisiä tunnuksia. Vastaussanoman kentässä 5 on yrityksen nimi, kentässä 8 on selväkielinen y-tunnus, kentässä 10 on selväkielinen yrityskäyttäjän henkilötunnus ja kentässä 11 on yrityskäyttäjän nimi.
09	Salattu Y-tunnus ja salattu yrityskäyttäjän henkilötunnus tai pankin ja palveluntuottajan keskenään sopima salattu muu tunnus. Pyydetty salattuja tunnuksia. Vastaussanoman kentässä 5 on yrityksen nimi, kentässä 8 on salattu y-tunnus, kentässä 10 on salattu yrityskäyttäjän henkilötunnus ja kentässä 11 on yrityskäyttäjän nimi.

Taulukko 7. Yksilöintitiedon koodimääritelmä 1Y. (FK 2011a, 17)

1Y	Selite – Kaikkien tai osa pyydetyistä tiedoista ei löytynyt
10	Pyydettyjä tietoja ei löytynyt asiakkaasta.
11	Yritysasiakkaan käyttäjästä ei löytynyt henkilötunnusta.
12	Yritysasiakkaasta ei löytynyt y-tunnusta.

Taulukon 4 kenttä 10. Käyttäjän tunnus ei ole pakollinen elementti vastaussanomassa. Pankki lähettää vastaussanomassa taulukon 4 kentän 10 B02K_USERID-elementin, jos vastaussanoman taulukon 4 kentän 9 B02K_CUSTTYPE-elementin arvo on 08 tai 09 (FK 2011a, 14). Pankki lähettää

yrityskäyttäjän henkilötunnuksen tai salatun tunnuksen, jos palveluntarjoaja on sitä pyytänyt tunnistuspyynnössä. Kaikki Suomen pankit eivät tarjoa yrityksille tupas-tunnistautumisen mahdollisuutta. (FK 2011a, 9)

Taulukon 4 kenttä 11. Yrityskäyttäjän henkilönimi ei ole pakollinen elementti vastaussanomassa. Taulukon 4 kentän 11 B02K_USERNAME-elementti on vastaussanomassa, jos vastaussanoman taulukon 4 kentän 9 B02K_CUSTTYPE-elementin arvo on 08 tai 09 (FK 2011a, 14). Pankki lähettää vastaussanoman B02K_USERNAME-elementissä yritysverkkopankkitunnuksen yrityskäyttäjän henkilönimen. (FK 2011a, 9)

Taulukon 4 kenttä 12. Vastaussanoman MAC-turvatarkiste, joka yksilöi asiakkaan ohjaaman url-osoitteen query string -merkkijonon (Nordea Pankki Suomi Oyj 2011, 10). MAC-turvatarkiste varmistaa query string -merkkijonon elementtien eheyden ja lähettäjän alkuperän, koska palveluntarjoajan yksilöivä merkkijono eli tarkisteavain on tiedossa vain palveluntarjoajalla ja pankilla. Tarkisteavaimen on mahdollista olla esim. LEHTI. (Nordea Pankki Suomi Oyj 2011, 12–13)

Vastaussanoman MAC-turvatarkiste muodostuu yhdistämällä taulukon 4 kentistä 1 – 9 tai kentistä 1 – 11 elementtien arvot merkkijonoksi. Jokaisen elementin jälkeen lisätään & -merkki. Merkkijonon loppuun lisätään tarkisteavain, jonka jälkeen lisätään & -merkki. Merkkijonon muodostamisen jälkeen merkkijonosta lasketaan hash-merkkijono. Hash-algoritmin valinta perustuu taulukon 4 kentän 7 B02K_ALG-elementin tyyppikoodista. Hash-algoritmista muodostunut heksadesimaalimerkkijonon A – F -kirjaimet muutetaan isoiksi kirjaimiksi. (FK 2011a, 11)

Taulukon 4 kentistä 1 – 9 muodostuu merkkijono, josta pankki muodostaa MAC-turvatarkisteen ja palveluntarjoaja tarkistaa MAC-turvatarkisteen.

B02K_VERS&B02K_TIMESTAMP&B02K_IDNBR&B02K_STAMP&
B02K_CUSTNAME&B02K_KEYVERS&B02K_ALG&B02K_CUSTID&
B02K_CUSTTYPE&tarkisteavain&

Taulukon 4 kentistä 1 – 11 muodostuva merkkijono, josta pankki muodostaa MAC-turvatarkisteen ja palveluntarjoaja tarkistaa MAC-turvatarkiseen. Liite 3 havainnollistaa MAC-turvatarkisteen tarkistamisen php-koodilla.

B02K_VERS&B02K_TIMESTAMP&B02K_IDNBR&B02K_STAMP&
B02K_CUSTNAME&B02K_KEYVERS&B02K_ALG&B02K_CUSTID&
B02K_CUSTTYPE&B02K_USRID&B02K_USERNAME&tarkisteavain&

Seuraavana on url-osoite, joka havainnollistaa tupas-tunnistautujan ohjaavan url-osoitteen rakenteen.

https://www.palveluntarjoaja.fi/tupas_hyvaksytty?B02K_VERS&B02K_TIMESTAMP&B02K_IDNBR&B02K_STAMP&B02K_CUSTNAME&B02K_KEYVERS&B02K_ALG&B02K_CUSTID&B02K_CUSTTYPE&B02K_USRID&B02K_USERNAME&B02K_MAC

Seuraavana on url-osoite, joka havainnollistaa tupas-tunnistautujan ohjauksen pankilta palveluntarjoajan internetsivulle. Url-osoite havainnollistaa query string -merkkijonon elementit. Tupas-tunnistautuja ei näe internetselaimessa ohjaavaa url-osoitteen query string -merkkijonoa, jos palveluntarjoaja on url-osoitteen query string -merkkijonon käsittelyn jälkeen ohjannut käyttäjän toiseen url-osoitteeseen.

https://www.palveluntarjoaja.fi/tupas_hyvaksytty?B02K_VERS=0002&B02K_TIMESTMP=20020111022160939 42&B02K_IDNBR=50178090&B02K_STAMP=20111022160940&B02K_CUSTNAME=NORDEA+%2F+DEMO&B02K_KEYVERS=0001&B02K_ALG=01&B02K_CUSTID=2102819988&B02K_CUSTTYPE=01&B02K_MAC=881B57D6F07270C58489FFAC44B6A400

3.6. Osapuolten vastuu

Pankki

Pankki on vastuussa virheellisten tietojen lähettämisestä palveluntarjoajalle. Pankki on velvollinen maksamaan korvausta palveluntarjoajalle palveluntarjoajan järjestelmälle aiheutuneista vahingoista tai liiketoiminnalle aiheutuneista vahingoista. Korvauksen summa vaihtelee pankin ja palveluntarjoajan välisen sopimuksen sisällön mukaan. (OP-Pohjola-ryhmä 2012, 1)

Palveluntarjoajan lähettäessä pankille virheellistä tietoa on pankki vastuussa, jos pankki hyväksyy vastaanotetut virheelliset tiedot. Pankin vastuu edellyttää, että palveluntarjoaja on lähettänyt virheellistä tietoa tahattomasti. Pankki on vastuussa palveluntarjoajan ja pankin välillä tapahtuneesta yhteysvirheestä, joka korruptoi tunnistautumistapahtuman tietoja, jos pankki käsittelee korruptoituneet tiedot. (OP-Pohjola-ryhmä 2012, 1)

Palveluntarjoaja

Palveluntarjoaja on vastuussa oman järjestelmänsä tietoturvasta. Palveluntarjoajan laiminlyödessä tietoturvaa on palveluntarjoaja vastuussa vahingoista. Palveluntarjoaja on vastuussa asiakkaasta arkaluonteisista tiedoista, jos pankin ja palveluntarjoajan välinen sopimus ei toisin määritä. (OP-Pohjola-ryhmä 2012, 1–2)

Henkilötietolain mukaisesti arkaluonteisten tietojen käytöstä on palveluntarjoajan ilmoitettava viranomaisille, jos pankilta pyydetään arkaluonteisia tietoja asiakkaasta tunnistautumisen yhteydessä. Palveluntarjoajan henkilötunnuksen käsittelemiseen pitää tehdä sopimus viranomaisten kanssa. (Tietosuojavaltuutetun toimisto 2012) Radiomikrofonisivustoa koskevista henkilötietolakiasioista on Sini Tanskanen tehnyt opinnäytetyön aiheesta Asiakasrekisteröintimoduuli Drupalissa 2012.

Pankki lähettää palveluntarjoajalle henkilötunnuksen vain, jos palveluntarjoajalla on oikeus rekisteröidä tiedot järjestelmään (Nordea Pankki Suomi Oyj 2011, 13). Asiakkaalle on ilmoitettava selkeästi, jos asiakkaasta pyydetään pankilta arkaluonteisia henkilötietoja (Tietosuojavaltuutetun toimisto 2012).

4 DRUPAL-SISÄLLÖNHALLINTAJÄRJESTELMÄ

4.1. Lisenssiehdot

Drupal on GPL-lisenssin alainen järjestelmä (About Drupal 2011). GPL-lisenssin lyhenne tarkoittaa Global Public Licence ja suomeksi yleinen lisenssi. GPL-lisenssin käyttö tarkoittaa, että kaikki ohjelmiston lähdekoodit on annettava asiakkaalle. Asiakkaalla on lupa muokata lähdekoodia sekä myydä ohjelmistoa eteenpäin. Palvelun käyttäjälle ei tarvitse antaa järjestelmän lähdekoodia. (Välimäki 2007)

GPL-lisenssi ei suojaa kaikkia Drupalin lähdekoodeja. GPL-lisenssi suojaa kaikki lähdekoodit, jotka käyttävät tai toimivuudeltaan ovat sidoksissa Drupalin lähdekoodiin. Esim. moduuli, joka käyttää Drupalin koodillisia funktiota toimiakseen, on GPL-lisenssin suojaamaa koodia. JavaScript-, jQuery-, HTML- ja normaali teksti eivät ole Drupalissa GPL-lisenssin suojaamaa koodia. JavaScript- ja jQuery-lähdekoodit, jotka käyttävät Drupalin JavaScript- tai jQuery-funktioita hyväkseen ovat GPL-lisenssin suojaamia. (About Drupal 2011)

Drupal tarjoaa Git-versiohallintapalvelun jokaiselle Drupal-moduuliprojektille. Kaikki Drupal Git-versiohallinnassa olevat lähdekoodit kuuluvat automaattisesti GPL-lisenssin suojaamaksi. (About Drupal 2011)

4.2. Moduulin koostumus

Hooks -järjestelmäfunktiot

Drupal perustuu hook-ajatusmalliin. Hook tarkoittaa PHP-funktiota, joka nimetään esim. hook_help(). Hook nimitys korvataan moduulin nimellä, jolloin funktio kirjoitetaan esimerkki_help(). Suomeksi hook-ajatusmallia voisi ajatella koukuksi, johon Drupal kiinnittyy prosessoinnin aikana. Moduulin tarvitsee vain implementoida hook-funktio saadakseen Drupal-ytimen toimintoja käyttöön. Drupal

tietää kaikkien moduulien implementoidut hook-funktiot. Drupal on kykenevä valitsemaan oikeanlaiset funktiot suoritettavaksi hook-logiikalla. (Drupal Community Documentation 2011a)

Moduulin tiedostot

Moduuli koostuu vähintään kahdesta tiedostosta, jotka ovat info- ja module-tiedostopäätteisiä. Info- ja module-tiedostojen nimien on oltava samannimiset kuin moduulin nimi. Muita tiedostopäätteisiä tiedostoja moduulilla on mahdollista olla engine, install, profile, test, php, inc, js ja css. Yleinen käytäntö on, että jokaisen tiedoston nimi alkaa moduulin nimellä. Moduulin kansion nimi kirjoitetaan pienillä kirjaimilla.

Info-tiedosto

Info-tiedosto sisältää kaikki tiedot moduulista, joita Drupalin tarvitsee tietää moduulin toimimiseksi. Vähimmäismääränä info-tiedostoon pitää merkitä moduulin nimi, lyhyt kuvaus moduulin tarkoituksesta, Drupal-ytimen version määritys ja module-tiedoston nimi. Info-tiedostossa rivin kommentointi on mahdollista puolipistettä käyttäen. Seuraavana on esimerkki info-tiedoston moduulin määritelmistä.

```
name = Esimerkki
description = Esimerkki opinnäytetyössä.
core = 7.x ;esimerkki kommentti
files[] = esimerkki.module
```

Info-tiedostoon on mahdollista merkitä, myös moduuli-paketin nimi, osoitepolun moduulin asetukset sivulle, moduulin versionumero ja riippuvaisuudet muista moduuleista. Muiden moduulien riippuvaisuus versionumerosta on mahdollista määrittää. Riippuvuus versiosta on määriteltävä tarkasti. Riippuvuutta ei ole mahdollista merkitä esim. 7.x. Riippuvuuden on mahdollista merkitä yhtäsuuruus- tai suurempi-operaattorimerkinnöillä. Moduulin riippuvuus erityiseen PHP-versioon on mahdollista määrittää erikseen tarkka PHP-versio. Seuraava esimerkki vastaa moduulin info-tiedostoa.

```
name = Esimerkki
description = Esimerkki opinnäytetyössä.
package = Esimerkit
core = 7.x
configure = admin/config/esimerkki
version = 7.x-0.1

php = 5.3.5
dependencies[] = user
dependencies[] = system (>=7.9)

files[] = esimerkki.module
files[] = esimerkki.install
```

Install-tiedosto

Install-tiedosto on tarpeellinen tiedosto moduuleille, jotka tarvitsevat käyttöönoton yhteydessä määrittelyjä tietokantaan tai Drupal-asetuksiin. Poistaessa moduulin Drupalista install-tiedostoon on mahdollista määrittää moduulin poistoon tarvittavia toimenpiteitä.

Install-tiedoston schema-funktiossa määritetään tietokantaan tietokantatauluja. Tietokantataulun sarakkeiden tyypeistä ja pituuksista on opinnäytetyön tietokantaosiosta lisää. Taulun tiedot määritellään muuttujaan puurakenteisesti taulukoihin.

```
function esimerkki_schema() {

  Taulun nimi määritetään schema-muuttujan alkionimeen.
  $schema['esimerkki'] = array(

    Määritetään taulun kommentti ja tarkoitus.
    'description' => format_string('Esimerkki taulu.'),

    Taulun kentät määritellään fields-alkion taulukkoon.
    'fields' => array(

      Tietokantataulun sarakkeen nimi
      'esimerkki_id' => array(
```

```
            Määritetään sarakkeen kommentti ja tarkoitus.
            'description' => format_string('Esimerkki ID'),
            Sarakkeen tyyppi.
            'type' => 'serial',

            Määritetään sarakkeen pituus.
            'size' => 'normal',

            Merkintä, onko luku aina positiivinen.
            'unsigned' => true,

            Merkintä, voiko kenttä olla tyhjä.
            'not null' => true,
        ),
    ),

    Määritys taulun sarakkeiden indeksoinneista.
    'indexes' => array(

      Indeksin nimi sisältää indeksoitavan sarakkeen nimen taulukossa.
      'kayttajat' => array('kayttaja_id'),
    ),

    Taulun pääavaimet määritetään primary key -taulukkoon.
    'primary key' => array('esimerkki_id'),

    Taulun viiteavaimet määritellään foreign keys -taulukon sisälle.
    'foreign keys' => array(

      Sarakkeen nimi, johon viiteavain tulee.
      'kayttajien_id' => array(

        Tietokantataulun sarakkeen viiteavaimen osoitettavan
        tietokantataulun nimi.
        'table' => 'users',

        Osoitettavan tietokantataulun sarakkeen nimi.
        'columns' => array('kayttaja_id' => 'uid'),
      ),
    ),
  );

  Taulun määritelmät palautetaan Schema API rajapinnan käsiteltäviksi.
  return $schema;
}
```

Install-tiedoston install-funktio suorittaa moduulin käyttöönottamiseen vaadittavat toimenpiteet. Seuraavaksi on esimerkki install-funktiosta, joka asettaa järjestelmävalvojan roolille luvan käyttää moduulin esimerkkiä ja ilmoittaa käyttäjälle moduulin asennuksen onnistuneen.

```
function esimerkki_install() {
  user_role_grant_permissions(3, array('access esimerkki'));
  drupal_set_message('Esimerkki, Moduulin asennus onnistui!');
}
```

Install-tiedoston uninstall-funktiolla on mahdollista esim. poistaa moduulin asettamia asetusmuuttujia tai muokata tietokantatauluja. Uninstall-funktioon ei tarvitse määrittää moduulin schema-funktiossa luotujen taulujen poistoja tietokannasta.

```
function esimerkki_uninstall() {
  variable_del('esimerkki');
}
```

Module-tiedosto

Moduulin ensisijaiset funktiot ohjelmoidaan module-tiedostoon. Drupal olettaa moduulin ensisijaisten hook-funktioiden sijaitsevan module-tiedostossa. Ensisijaisia funktioita ovat esim. hook_help, hook_permission ja hook_menu.

Help-funktiossa määritellään moduulin ohjeistus. Drupalin käyttäjä voi katsoa moduulin käyttöohjeistukset Drualin Dashboardin kautta. Help-funktiolla voi palauttaa html-koodia. Help-funktion parametrina tulee osoitepolku, jonka avulla funktion sisällä määritetään oikea ohjeistus palautettavaksi. Moduulilla voi olla useita ohjeistussivuja.

```
function esimerkki_help($polku) {
  if ($polku == 'admin/help#esimerkki') {
    return t('Esimerkki ohjeistussivusta.');
  }
}
```

Moduuliin on mahdollista määritellä käyttäjäroolikohtaisia oikeuksia. Käyttäjäroolikohtaiset oikeusasetukset tulevat näkyviin Drupalin Dashboardin kautta People > Permissions. Permission-funktion tarkoitus on määrittää oikeudet käyttäjärooleille, mutta ei anna- eikä aseta oikeuksia käyttäjärooleille. Käyttäjäroolit moduulin määrittämiin käyttörajoituksiin on asetettava Drupalin käyttäjäroolihallintasivulta tai user_role_grant_permissions-funktiolla.

```
function esimerkki_permission() {
  return array(
    'access esimerkki' => array(
      'description' => t('Oikeus avata esimerkkisivuja.'),
      'title' => t('Pääsy esimerkkisivuille'),
      'restrict access' => TRUE,
    ),
  );
}
```

Hook_menu-funktiossa määritetään url-osoite, jossa moduulin on tarkoitus aktivoitua. Määritelty url-osoite aktivoi moduulissa määritetyn funktion tai Drupal-ytimessä olevan funktion. Hook_menu-funktiolla on mahdollista määrittää Drupal navigation blockiin uusia Drupal-internetsivuston sisäisiä url-linkkejä. Seuraavaksi on esimerkki_menu-funktiosta, joka havainnollistaa url-osoitteen määrittämisen. (Tomlinson & VanDyk 2010, 57)

```
function esimerkki_menu() {
  $sivu['esimerkki/esimerkillinen'] = array(
    'title' => 'Esimerkkillinen',
    'page callback' => 'esimerkki_funktio',
    'file' => 'esimerkki_esimerkkifunktiot.inc',
    'access callback' => 'user_access',
    'access arguments' => array('access esimerkki'),
    'type' => MENU_NORMAL_ITEM,
    'weight' => '-10',
  );
  return $sivu;
}
```

Drupal-internetsivuston sisäinen osoitepolku määritetään sivu-muuttujan alkioon. Title-alkio sisältää linkin näkyvän tekstin ja on vaihtoehtoinen. Page callback -alkioon määritetään funktio, joka suoritetaan url-osoitteessa. File-alkioon määritetään suoritettavan funktion tiedoston nimi, joka vastaa php include_once-funktiota. Tiedoston määrittäminen on vaihtoehtoinen. (Tomlinson & VanDyk 2010, 61)

Access callback -alkioon määritetään käyttäjän oikeudet suoritettavaan funktioon tai internetsivulle. Access callback on vaihtoehtoinen. Access arguments -alkioon määritetään taulukkoon käyttäjärooleille asetettavan funktion suoritus tai internetsivulle pääsyn oikeuden nimitys. Oikeuden nimitys on määritetty hook_permission-funktiossa. Access callbackin ja access argumentsin määrittäminen ovat vaihtoehtoisia. (Tomlinson & VanDyk 2010, 61)

Type-alkioon määritetään url-osoitteen näkyvyys internetsivustolla. MENU_CALLBACK ilmaisee, että moduuli aktivoi määritetyssä url-osoitteessa page callback -alkiossa määritellyn funktion. MENU_NORMAL_ITEM asettaa url-linkin Drupalin menu-listaan, joka tulee näkyville navigation blockissa. Weight-alkio määrittää näkyvyyden paikan navigation blockissa. Näkyvyyden painoitus on mahdollista määrittää -100 - +100 välille. Weight-alkion määrittäminen on vaihtoehtoinen. (Tomlinson & VanDyk 2010, 61–62)

4.3. Tietokanta

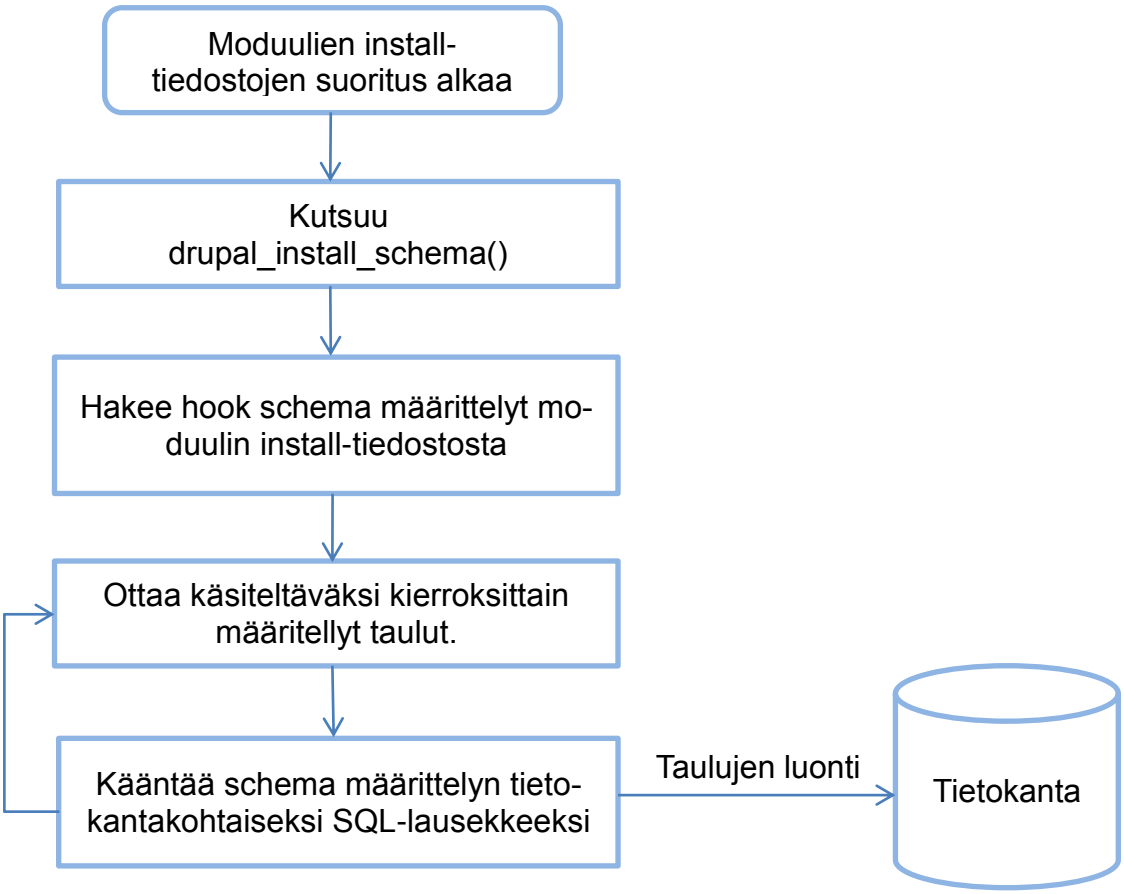
Schema API

Schema API on tietokantarajapinta usealle tietokantajärjestelmälle. Schema API määrittää Drupal-standardin, jonka avulla moduulien kehittäjät määrittävät tietokantatoiminnot, jotka toimivat useaan tietokantajärjestelmään. Schema API sisältää rajapinnan, jonka avulla on mahdollisuus ohjelmoida räätälöity lähdekoodikirjasto tietokantajärjestelmälle. Drupal-ytimen standardiasennus sisältää MySQL-, PostgreSQL- ja SQLite-tietokantarajapinnat. Lisäasennuksena on saa-

tavilla MS SQL- ja Oracle-tietokantarajapinnat. (Drupal Community Documentation 2011b)

Schema API -tietokantarajapinta tuo moduulien kehittäjille työkalun suunnata moduulin toimivuus usealle tietokantajärjestelmälle. Esim. MySQL-tietokantajärjestelmän sijaan valitaan Oracle-tietokantajärjestelmä. Tietokantajärjestelmien huomioonottaminen ja moduulien räätälöinti on huomattavasti vähäisempää Schema API rajapintaa käyttäessä. Drupalilla on aina tieto Drupalin tietokannassa olevista tietokantatauluista, jos tietokannan sisältö on tuotettu Drupalin Schema API -rajapintaa hyväksikäyttäen. (Drupal Community Documentation 2011b)

Seuraavana on kuvio 3, joka havainnollistaa moduulin hook_schema-funktiossa määritettyjen tietokantataulujen suoritusjärjestyksen moduulin asennuksen yhteydessä.



Kuvio 3. Moduulien hook_schema suoritus. (Tomlinson & VanDyk 2010, 101)

Schema API muodostaa SQL-lauseen useana kierroksena. Yksi kierros on hook_schema-funktion schema-muuttujassa oleva alkion käsittely. Schema API muuttaa tietokantataulun määritykset tietokantakohtaiseksi SQL-lauseeksi. Tietokantataulujen määrittelyt muodostettua SQL-lauseeksi, Schema API lähettää SQL-lauseen tietokannalle esim. MySQL.

Taulukko 8 sisältää Schema API:n sisältämät tietokantadatatyypit. Schema API rajapinta sisältää valmiiksi ohjelmoituna taulukossa 8 mainitut datatyypit. Taulukko ilmaisee Drupal Schema API -datatyypin nimikkeen. MySQL- ja PostgreSQL-datatyypit poikkeavat toisistaan. Schema API valitsee sopivan datatyypin käytössä olevaan tietokantaan. Schema API ei sisällä kaikkia tietokantajärjestelmien datatyyppejä.

Taulukko 8. Schema API tietokantadatatyypit. (DCD. 2011a)

Tyyppi	Koko	MySQL *)	PostgreSQL *)	SQLite tyyppi
serial	tiny	tinyint, 1 t	serial, 4 t	integer
serial	small	smallint, 2 t	serial, 4 t	integer
serial	medium	mediumint, 3 t	serial, 4 t	integer
serial	normal	int, 4 t	serial, 4 t	integer
serial	big	bigint, 8 t	bigserial, 8 t	integer
int	tiny	tinyint, 1 t	smallint, 2 t	integer
int	small	smallint, 2 t	smallint, 2 t	integer
int	medium	mediumint, 3 t	int, 4 t	integer
int	normal	int, 4 t	int, 4 t	integer
int	big	bigint, 8 t	bigint, 8 t	integer
float	tiny	float, 4 t	real, 6 numeroa	float
float	small	float, 4 t	real, 6 numeroa	float
float	medium	float, 4 t	real, 6 numeroa	float
float	normal	float, 4 t	real, 6 numeroa	float
float	big	double, 8 t	double precision, 15 numeroa	float
numeric	normal	numeric, 65 numeroa	numeric, 1000 numeroa	numeric

*) tyyppi ja koko/leveys (jatkuu)

Taulukko 8 (jatkuu).

Tyyppi	Koko	MySQL *)	PostgreSQL *)	SQLite tyyppi
varchar	normal	varchar, 255 t tai 64 Kt MySQL 5.0.3 ->	varchar, 1 Gt	varchar
char	normal	char, 255 t	character, 1 Gt	Ei tue
text	tiny	tinytext, 256 t	text, rajoittamaton	text
text	small	tinytext, 256 t	text, rajoittamaton	text
text	normal	text, ~64 Kt	text, rajoittamaton	text
text	medium	mediumtext, ~16 Mt	text, rajoittamaton	text
text	big	longtext, ~4 Gt	text, rajoittamaton	text
blob	normal	blob, ~64 Kt	bytea, 4 Gt	blob
blob	big	longblob, ~4 Gt	bytea, 4 Gt	blob
datetime	normal	datetime, vuodesta 1001 vuoteen 9999	timestamp, vuodesta -4713 vuoteen 5874897	Ei tue

*) tyyppi ja koko/leveys

Schema API ei sisällä aikaleimaista datatyyppiä. Aikaleima on aikavyöhykkeeseen sidoksissa. Tietokannan siirtäessä palvelimelta toiseen aikavyöhykkeelliseen palvelimeen muuttuu aika epätarkaksi. Erilaiset tietokantajärjestelmät saattavat huomioida aikavyöhyketiedon. Tietokantariippuvaiset datatyypit hankaloittavat tietokannan saumatonta siirtämistä esim. MySQL-tietokannasta PostgreSQL-tietokantaan. Schema API rajapinta ei sisällä toimintoja, jotka eivät mahdollista saumattoman tietokantariippumattomuuden (Drupal Community Documentation 2011b). (Billauer 2009)

Tietokantadatatyypit, joita Schema API ei sisällä, on mahdollista määrittää erikseen hook_schema-funktiossa. Hook_schema-funktiossa määritetylle sarakkeelle on mahdollista määrittää samanaikaisesti usealle tietokannalle tietokantakohtaisia määritelmiä. Seuraavaksi havainnollistava esimerkki, jossa määritetään datatyyppi tietokantakohtaisesti MySQL-tietokantaan. (Tomlinson & VanDyk 2010, 107)

```
$schema['esimerkki'] = array(
  'fields' => array(
    'esimerkki_sarake' => array(
      'description' => format_string('Esimerkki datatyyppi.'),
      Schema-API valitsee tämän, jos MySQL on käytössä.
      'mysql_type' => 'TINYBLOB',
      Schema-API valitsee tämän, jos on jokin muu tietokanta käytössä.
      'type' => 'blob',
      Schema-API valitsee tämän, jos on jokin muu tietokanta käytössä.
      'size' => 'normal',
    ),
  ),
);
```

Schema API määrittää pääavaimen tietokantatauluun. Schema API sisältää rakenteen viiteavaimen määrittämiseen, mutta jättää huomiotta viiteavaimen määrittelyt moduulin asennuksen yhteydessä. Moduulin hook_install-funktioon on määriteltävä SQL-lauseella erikseen taulumuunnos viiteavaimelle. (Frando 2010)

Database API

Database API on rajapinta, joka kääntää rajapinnalla ohjelmoidut tietokantatoiminnot tietokantakohtaiseksi SQL-lauseeksi. Database API -rajapinnan tarkoitus on nopeuttaa moduulien kehitystä. Tietoturva on parempi Database API -rajapintaa käytettäessä, koska tietokantaa vahingoittavia SQL-syntaksivirheitä ei ilmene. Database API muodostaa moduuleille rakenteellisen standardin tietokantakyselyille. Database API on oliopohjaisesti ohjelmoitu. Moduulin kehittäjällä on mahdollista käyttää rajapintaa oliopohjaisesti ja proseduraalisesti. (DCD 2011b)

Db_query-funktio on yksinkertaisille tietokantakyselyille suunniteltu. Funktioon määritetään parametrina SQL-lause. Db_query-funktioon määritettyyn SQL-lauseeseen ei kirjoiteta SQL-lauseen päättävää puolipistettä.

```
$tulos = db_query('SELECT esimerkki_id FROM esimerkki');
```

Db_select-funktio on suunniteltu tietokantakyselyille, jotka ovat pitkiä ja monimutkaisia. Db_select-funktioon määritetään parametrina tietokantatauluntaulun nimi ja lyhenne. Fields-funktiolla määritetään sarakkeet, jotka tietokanta palauttaa. Parametrina fields-funktioon määritetään taulun lyhenne ja sarakkeiden nimet, jotka sijoitetaan taulukkoon. AddField-funktiota on tarkoitus käyttää, jos haluaa määrittää jokaisen sarakkeen erikseen funktioittain. AddField-funktiolla määritetään parametrina taulun lyhenne, sarakkeen nimi.

Condition-funktio vastaa SQL-lauseen where-määritystä. Parametrina condition-funktioon määritetään sarakkeen nimi, verrattava arvo ja operaattori tai monimutkaisempi IN, LIKE, BETWEEN. Condition-funktiossa määriteltyyn sarakkeen nimeen on mahdollista määrittää tietokantataulun lyhenteen sarakkeen nimen alkuun, joka on vaihtoehtoinen määritys. Execute-funktio aloittaa tietokantakyselyn suorittamisen.

Seuraavana on esimerkki koodi db_select-funktion käytöstä. Ensin määritetään db_select-funktioon ensimmäisenä parametrina tietokantataulun nimi ja toisena parametrina määritetään tietokantataululle lyhenne. Fields-funktioon määritetään ensimmäisenä parametrina tietokantataulun lyhenne ja toisena parametrina tietokantataulusta haettavien sarakkeiden nimet sijoitettuna taulukkoon. Condition-funktioon määritetään rajoitteet haulle. Condition-funktion ensimmäisenä parametrina määritetään sarakkeen nimi, jonka alkuosaan määritetään tietokantataulun lyhenne. Toisena parametrina määritetään sarakkeen verrattava arvo ja kolmantena parametrina määritetään operaattori. <>-merkintä tarkoittaa, että verrattava sarake ei saa olla arvoltaan esim. Firco. Range-funktioon määritetään rivien määrän, jonka tietokantajärjestelmä palauttaa. Execute-funktio suorittaa kyselyn.

Esimerkki db_select-funktion käyttö palauttaa tulos-muuttujaan esimerkki radiomikrofonit tietokantataulusta kaikkien radiomikrofoni_id-, valmistaja- ja mallitiedot joiden valmistaja sarake ei ole arvoltaan Firco. Db_select-funktio palauttaa oliopohjaisesti tiedot.

```
$tulos = db_select('radiomikrofonit', 'r')
->fields('r', array(radiomikrofoni_id', 'valmistaja', 'malli'))
->condition('r.valmistaja', 'Firco', '<>')
->range(0, 500)
->execute();
```

Tietokantakyselyn ollessa kahden taulun liitos, muuttuu db_select-funktiolla tietokantakyselyn määrittämisen rakenne. Tietokantakyselyssä tauluja yhdistämistä varten on db_select-funktio asetettava kysely muuttujaan. Seuraavaksi havainnollistava esimerkki db_select-funktion käytöstä.

```
$kysely = db_select('esimerkki', 'e');
$kysely
->fields('e', array('esimerkki_id', 'etunimi', 'sukunimi'))
->join('palkka', 'p', 'e.esimerkki_id = p.esimerkki_id');
$kysely->condition('p.palkka', 5000, '>');
$kysely
->addJoin('INNER', 'bonustaso', 'b', 'p.bonus_id = b.bonus_id');
$kysely->addField('p', 'palkka');
$kysely->addField('b', 'bonus_nimi');
$kysely->orderBy('e.sukunimi', 'ASC');
$tulos = $kysely->execute();
```

Tietokantakyselyn kohdistuessa useampaan kuin kahteen tauluun on jokainen lisätaulu liitettävä addJoin-funktiolla tai lisäämällä join-funktio. Join-funktion oletuksena on inner-tauluyhdistys. AddJoin-funktioon määritetään taulujen yhdistämisen suunta, jotka ovat INNER, LEFT OUTER ja RIGHT OUTER. OrderBy-funktiolla on mahdollista järjestää tuloksen järjestys. OrderBy-funktion parametrina määritetään sarakkeen nimi ja järjestyksen suunta ASC tai DESC.

Tietokantatauluun asetetaan uusi rivi db_insert-funktiolla. Db_insert-funktion parametrina määritetään taulun nimi. Fields-funktioon parametrina määritetään taulukkoon sarakkeiden nimet. Values-funktioon parametrina määritetään tau-

lukkoon sarakkeiden nimet, joihin asetetaan tauluun asetettavat arvot. Useita values-funktioita on mahdollista määrittää. Samaan values-funktioon parametrina taulukon sisälle on mahdollista määrittää useita asetettavia rivejä. Db_insert-funktio palauttaa lisätyn rivin numeraalisen tunnuksen.

```
db_insert('esimerkki')
->fields(array(
  'etunimi',
  'sukunimi'
))
->values(array(
  'etunimi' => 'Kalle',
  'sukunimi' => 'Kangas'
),
array(
  'etunimi' => 'Milla',
  'sukunimi' => 'Virtanen'
))
->execute();
```

Tietokantatauluun asetettaessa db_select-funktion SQL-kyselyllä, käytetään from-funktiota. From-funktioon asetetaan parametrina SQL-kyselyn muuttuja, joka sisältää db_select-funktion määritelmät.

```
db_insert('esimerkki')
->from($kysely)
->execute();
```

Tietokantataulun päivitys määritetään db_update-funktiolla. Db_update-funktion parametrina määritetään päivitettävän taulun nimi. Fields-funktioon parametrina määritetään taulukkoon päivitettävän sarakkeen nimi, johon asetetaan päivitettävä arvo. Condition-funktioon parametrina määritetään sarakkeen nimi, joka on päivityksen ehto. Toisena parametrina condition-funktioon määritetään vertailtava arvo ja kolmanteen parametriin määritetään vertailun operaattori.

```
db_update('esimerkki')
->fields(array(
  'etunimi' => 'Maija'
))
->condition('sukunimi', 'Kalle', 'LIKE')
->execute();
```

4.4. Tietoturva

Syötteiden tarkistus

Syötteet ovat web-sovellukselle tietoturvariski. Data, joka tulee tai lähtee pois järjestelmästä tuottaa mahdollisuuden tietoturvariskille. Web-sovellus, joka ei sisällä asianmukaisia syötteiden tarkistuksia, on tietoturvariski järjestelmälle ja palvelun käyttäjille. Drupal sisältää funktioita syötteiden tarkistukseen, jotka estävät web-sovellukseen, selaimeen ja tietokantaan kohdistuneita hyökkäyksiä. Merkkijonot, jotka saattavat sisältää ennalta tietämättömiä erikoismerkkejä tai merkkijonoyhdistelmiä on tarkistettava ja suodatettava. (Auger 2011)

Drupal tarjoaa merkkijonossa olevien html-koodien poistamiseen funktioita. T-, format_string- ja check_plain-funktiot tarkistavat ja suodattavat html-koodin merkkijonosta, muuttaen html-elementit html-kirjaimiksi. Edellä mainitut funktiot kutsuvat toisiaan. T-funktio palauttaa format-string-funktion suorituksen, jonka sisällä suoritetaan check_plain-funktio.

T-funktio on suunniteltu merkkijonojen kielikäännösten muodostamiseen. Esimerkiksi englanninkielinen merkkijono käännetään suomeksi. Drupal tekee kielimuunnoksen vertailemalla tietokannassa olevia sanoja merkkijonossa oleviin sanoihin. T-funktion ensimmäisenä parametrina ei saa parametrina asettaa muuttujaa. Muuttujat määritetään toiseksi parametriksi taulukkoon sijoitettuna. Taulukossa olevat muuttujat määritetään merkkijonoon paikkamerkeillä, jotka vastaavat taulukon alkioiden nimeä. T-funktiota ei saa käyttää hook_schema-funktion tietokantataulujen määrittelyssä (Webchick 2008).

```
t('Esimerkki @etunimi ja @sukunimi',
array('@etunimi' => $etunimi, '@sukunimi' => $sukunimi));
```

Format_string-funktio on parametrin toimivuudeltaan samanlainen kuin t-funktio. Format_string-funktiota käytetään t-funktion sijaan, jos merkkijonoa ei käännetä. Format_string-funktio tarkistaa parametrina saadun taulukon, joka sisältää merkkijonoon tulevat muuttujat. Tarkistettavan muuttujan alkionimen ensimmäinen merkki ilmaisee tarkistustavan. !-merkki ilmaisee, että muuttujaa ei tarkisteta. @-merkki ilmaisee, että muuttuja tarkistetaan check_plain-funktiolla. Prosenttimerkki % ilmaisee, että muuttuja suoritetaan drupal_placeholder-funktiolla, joka lisää -elementin muuttujan ympärille ja suorittaa check_plain-funktion muuttujalle.

```
$auto = '<h1>Audi</h1>';
format_string('Esimerkki @auto', array('@auto' => $auto));
```

Esimerkki check_plain-funktio palauttaa format_string-funktiolle <h1>Audi</h1> merkkijonon.

Drupal sisältää XSS-suodatuksen. XSS-lyhenne tulee Cross Site Scripting nimityksestä (RSnake 2011). Filter_xss-funktio tarkistaa merkkijonosta kaikki ei halutut html-elementit ja tarkistaa, että elementit on oikein määritetty. Funktio poistaa html linkin, joka sisältää JavaScriptiä.

Seuraavana on esimerkki filter_xss-funktiosta. Ensimmäisenä parametrina määritetään muuttuja, joka sisältää tarkistettavan merkkijonon. Toisena parametrina määritetään taulukkoon html-elementit, jotka filter_xss-funktio jättää merkkijononoon muuttumattomana.

```
filter_xss($merkkijono,
$hyvaksytyt_elementit = array('a', 'em', 'strong', 'ul', 'li'));
```

Form API

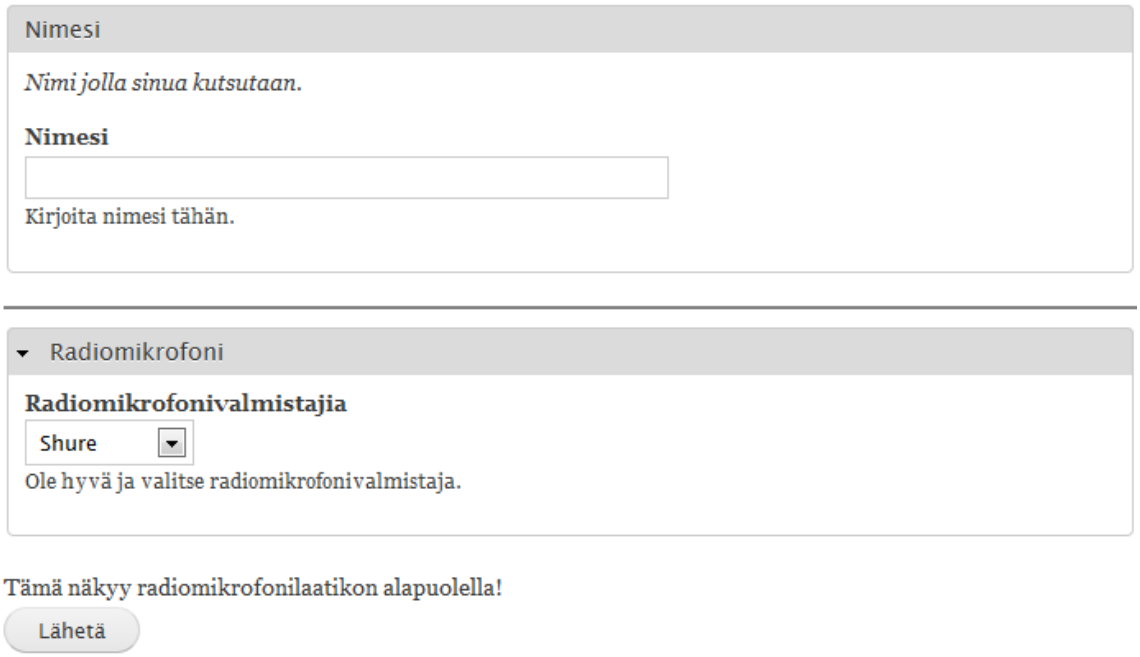
Form API on rajapinta, jonka avulla Drupal muodostaa ja käsittelee tietoturvallisesti html form -elementtejä (DCD 2008). Form API luo html form -elementtien määrittelystä html-koodin ja yksilöi html form -lomakkeen, sekä lisää kolme yksilöivää html form -lomake-elementtiä. Moduulissa olevan funktion nimellä ei ole väliä, jossa html form -lomakkeen elementit määritetään. Lomake määritetään taulukkoon puurakennemaisesti. Lomaketta määritettäessä on mahdollista jättää määrittelemättä taulukon alkioita tai jättää määritelmä tyhjäksi.

Lomakkeen määrittelytaulukko on kaksitasoinen. Ensimmäinen taso määrittelee alustan form-elementeille esim. html fieldset -elementti. Toinen taso määrittelee form-elementin alustaan esim. textinput-elementti. Title-alkioon on mahdollista määrittää elementin otsikko, joka tulee näkyville elementin yläpuolelle. Type-alkio määrittää html form -lomakkeen elementin tyypin. Description-alkioon on mahdollista määrittää html form -elementin tarkoitus. Tarkoitus tulee näkyville ylimmäksi alustan sisällä, jos kyseessä on ensimmäisen tason määritys. Toisen tason määritys tulee näkyville html form -lomakkeen elementin alapuolelle.

Prefix-alkioon on mahdollista määrittää ennen html form -lomakkeen elementtiä muodostuvia html-elementtejä. Suffix-alkioon on mahdollista määrittää html form -lomakkeen elementin jälkeen muodostuvia html-elementtejä. Collapsible-alkioon on mahdollista määrittää, onko alusta kokoon taittuva. Collapsed-alkioon määritetään, onko alusta kokoon taitettuna sivuston internetsivun latautuessa. Value-alkioon määritetään html form -elementin arvot. Options-alkioon määritetään näkyvät arvot, jos kyseessä on dropdownlist html form -lomakkeen elementti.

Seuraavana on havainnollistava php-koodiesimerkki Form API rajapinnalla määritellystä html form -lomakkeesta. Kuva 1 havainnollistaa Form API -rajapinnalla määritellyn html form -lomakkeen näkymän käyttäjän internetselaimessa.

```
function esimerkki_radiomikrofonin_valinta_form() {
$form['nimi'] = array(
    '#title' => t('Nimesi'),
    '#type' => 'fieldset',
    '#description' => t('Nimi jolla sinua kutsutaan.')
);

$form['nimi']['kutsumanimi'] = array(
    '#title' => t('Nimesi'),
    '#type' => 'textfield',
    '#description' => t('Kirjoita nimesi tähän.')
);

$form['radiomikrofoni'] = array(
    '#prefix' => '<hr />',
    '#title' => t('Radiomikrofoni'),
    '#type' => 'fieldset',
    '#collapsible' => TRUE,
    '#collapsed' => FALSE,
    '#suffix' =>
    '<div>'. t('Tämä näkyy radiomikrofonilaatikon alapuolella!')
    .'</div>',
);

$form['radiomikrofoni_vaihtoehdot'] = array(
    '#type' => 'value',
    '#value' => array(t('Shure'), t('Sennheiser'))
);

$form['radiomikrofoni']['valittu_radimikrofoni'] = array(
    '#title' => t('Radiomikrofonivalmistajia'),
    '#type' => 'select',
    '#description' =>
    t('Ole hyvä ja valitse radiomikrofonivalmistaja.'),
    '#options' => $form['radiomikrofoni_vaihtoehdot']['#value']
);

$form['submit'] = array(
    '#type' => 'submit',
    '#value' => t('Lähetä')
);


return $form;

}
```



Kuva 1. Form API esimerkki html form -lomake.

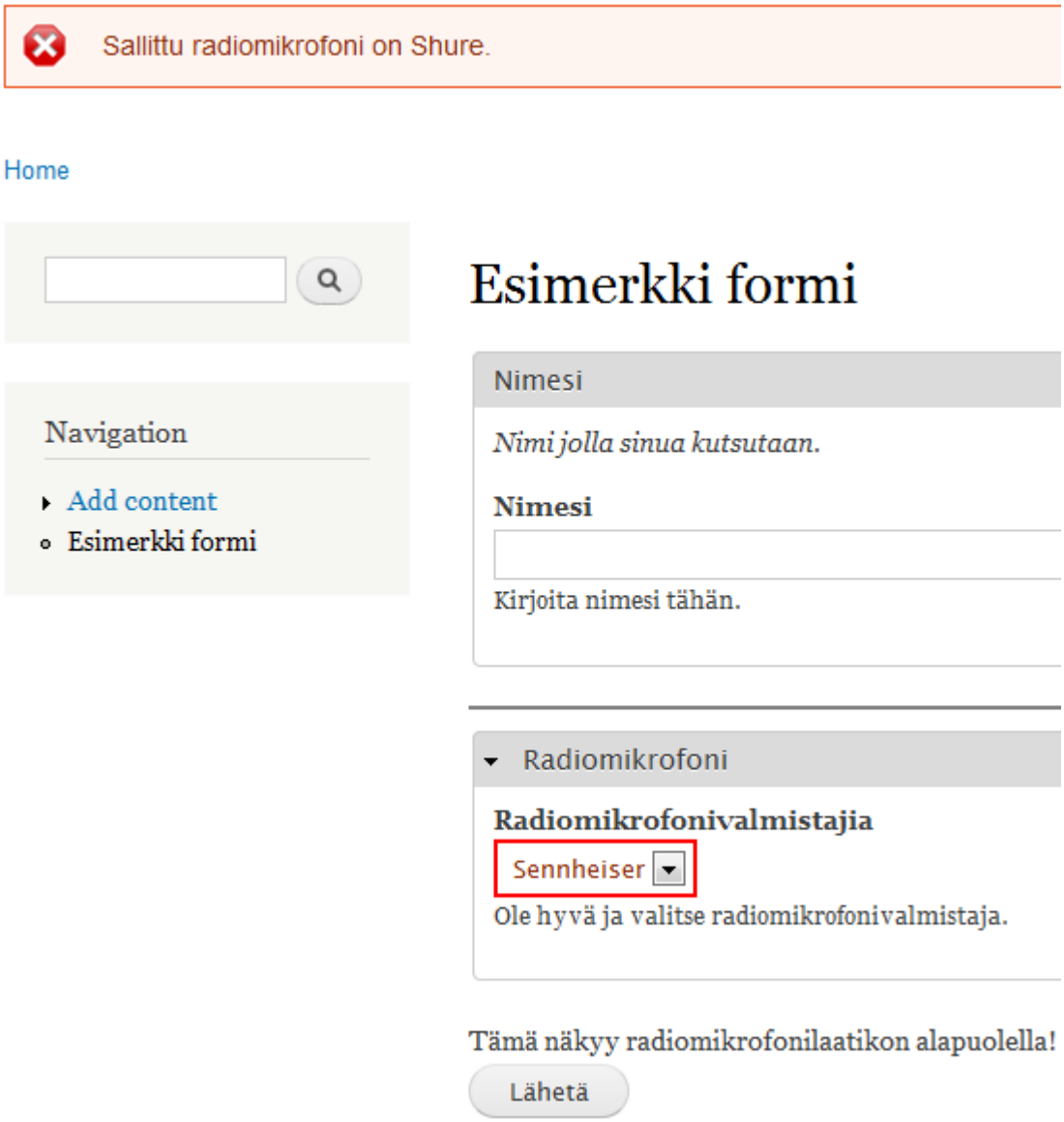
Html form -lomakkeen kolme yksilöivää elementtiä näyttävät selaimen sivuston-lähdekoodissa seuraavalta.

```
<input type="hidden" name="form_build_id"
  value="form-3CXJNmiU7qYmMRIz6gtxXBF26wxBfpSgTZcdB7Q96iU" />
<input type="hidden" name="form_token"
  value="ic5jWVALWz2I3po4WwsBkgK704fqsbEGVxvFRUrKSKw" />
<input type="hidden" name="form_id"
  value="esimerkki_autonvalinta_form" />
```

Lomakkeen tarkistus- ja prosessointi-funktion nimi määräytyy form-elementtien määrittelyfunktion nimestä. Funktio saa parametrina form_state-muuttujan, joka sisältää käyttäjän lähettämän lomakkeen form-elementtien arvot. Tarkistuksessa havaittu virhe on mahdollista ilmoittaa form_set_error-funktiolla käyttäjälle internetsivulle. Form_set_error-funktion ensimmäisenä parametrina asetetaan virheellisen html form -lomakkeen elementin nimi ja toisena parametrina virheilmoitus.

```
function esimerkki_radiomikrofonin_valinta_form_validate($form, &$form_state)
{
  $valinta = $form_state['values']['valittu_radimikrofoni'];

  if ($form_state['values'][rm_vaihtoehdot'][$valinta] != 'Shure') {
    form_set_error('valittu_radimikrofoni',
    t(Sallittu radiomikrofoni on Shure.'));
  }
}
```

Seuraavana on kuva 2, joka havainnollistaa Drupalin ilmoittaman virheen käyttäjälle internetsivulle ja kohdistaa punaisella neliöllä virheen kohdan html form -lomakkeessa.

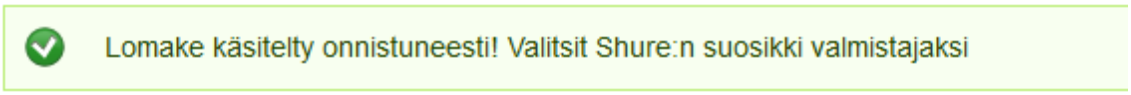


Kuva 2. Lähetetyn lomakkeen form-elementtien tarkistusesimerkki.

Submit-funktio suoriutuu, jos validate-funktion tarkistuksessa ei löytynyt virheitä. Parametrina tulee form_state-muuttuja, joka sisältää lähetetyn lomakkeen html form - lomakkeen elementtien arvot. Submit-funktio ohjaa käyttäjän oletuksena takaisin lomakkeen internetsivulle.

Kuva 3 havainnollistaa käyttäjän lähetettyä html form -lomakkeen Drupalin ilmoittavan viestin käyttäjälle. Kuva 3 mukainen ilmoitus ilmenee käyttäjälle, kun html form -lomake on läpäissyt hook_validate-funktion ja suoriutunut hook_submit-funktiosta.

```
function esimerkki_autonvalinta_form_submit($form_id, &$form_state) {
  $valinta = $form_state['values']['valittu_radimikrofoni'];
  $mikrofoni = $form_state['values']['radiomikrofoni_vaihtoehdot'][$valinta];
  drupal_set_message(t('Lomake käsitelty onnistuneesti!
  Valitsit @mikrofoni:n suosikki valmistajaksi',
  array('@mikrofoni' => $mikrofoni)));
}
```



Kuva 3. Lähetetyn lomakkeen form-elementtien prosessointiesimerkki.

Drupal tarjoaa funktion, joka näyttää viestin, jos käyttäjälle halutaan näyttää viesti internetsivun yläosaan internetsivun latautuessa. Viestin asettajafunktion nimi on drupal_set_message. Kuva 4 havainnollistaa Drupalin muodostavan viestin käyttäjälle.

```
drupal_set_message(t('Esimerkki, Moduulin asennus onnistui!'));
```

Kuva 4. Drupal set message -funktion viestiesimerkki.

Käyttäjien hallinta

Drupal tarjoaa käyttäjien hallintaan funktioita ja tarkistettavia muuttujia. Moduuleihin ja templateihin on mahdollista määrittää rajoituksia käyttäjille. Käyttäjien kirjautumistilan tarkistamiseen on user_is_logged_in-funktio, joka palauttaa boolean-tyyppisen arvon. Hook_permission-funktiolla käyttäjärooleille määritetään käyttäjäroolirajoituksia. Tarkemmin hook_permission-funktiosta opinnäytetyön moduulien koostumuksessa.

Käyttäjän roolitason tarkistamiseen on käytettävissä user_access-funktio. Parametrina funktioon annetaan tarkistettavan kohteen merkkijono. Funktio palauttaa boolean-tyyppisen arvon.

```
user_access('access esimerkki');
```

Käyttäjäroolitasolle luvan antaminen onnistuu user_role_grant_permissions-funktiolla. Ensimmäisenä parametrina määritetään käyttäjäroolitason id-tunnus. Toisena parametrina määritetään annettavan luvan kohteen nimi. Käyttäjäroolilta luvan muuttaminen onnistuu user_role_change_permissions-funktiolla. Käyttäjäroolilta luvan poistaminen onnistuu user_role_revoke_permissions-funktiolla.

```
user_role_grant_permissions(3, array('access esimerkki'));

user_role_change_permissions(2, array('access esimerkki'));

user_role_revoke_permissions(2, array('access esimerkki'));
```

Drupal-ytimen standardiasennus sisältää Trigger-moduulin. Trigger-moduulilla on mahdollista suorittaa toiminto jonkin tapahtuman seurauksena. Esim. käyttäjä rekisteröityy web-palveluun ja Trigger-moduuli estää automaattisesti uuden käyttäjätilin käyttämisen web-palvelussa.

Template-tasolla on mahdollista tarkistaa käyttäjän kirjautumisen logged_in-muuttujalla, joka on boolean-tyyppinen muuttuja. Sivuston ylläpitäjäroolin tarkistus onnistuu is_admin-muuttujalla, joka on boolean-tyyppinen muuttuja.

Järjestelmätason tietoturva

Drupal muuttaa sivuston suorituksen aikana PHP-asetuksia ini_set-funktiolla. Suorituksenaikaisia muutoksia Drupal muuttaa istunnon ja selaimen evästeen maksimi aikapituuden. Istunnon aikapituus Drupal-ytimen standardiasennuksessa on 30 minuuttia ja evästeen aikapituus on 1 667 minuuttia. Kaikki ini_set-funktiot eivät toimi oletuksena, jonka pitää tarkistaa PHP-asetuksista.

```
ini_set('session.gc_maxlifetime', 1800);

ini_set('session.cookie_lifetime', 100000);
```

Drupal on riippuvainen PHP-roskienkeruusta. Drupal muuttaa ajonaikana PHP-roskienkeruun todennäköisyyden ykköseksi ja roskienkeruun jakajaluvun ykköseksi. Todennäköisyys on roskienkeruuseen 1/1 eli 100 %.

```
ini_set('session.gc_probability', 1);
ini_set('session.gc_divisor', 1);
```

Drupal säilyttää Drupal-ytimen standardiasennuksessa settings.php-tiedostossa yksilöivän hash-suolausmerkkijonon. Suolaus nimitys tulee englanninkielisestä sanasta salt, jota käytetään hash-algoritmin yhdensuuntaistukseen (Salt cryptography 2011). Drupal suosittelee hash-suolausmerkkijonon säilyttämisen erillään sivuston asennuksesta ja varmuuskopioitavista kansioista. Drupal käyttää hash-suolausmerkkijonoa esim. form-lomakkeen, kirjautumislinkin ja peruutuslinkin yksilöimisessä.

```
$drupal_hash_salt = 'Jb4rO7c_r3oOqWSDAM5jxc9oACPiJ4uIbag9A9fp5CI';
```

Drupal-ytimen standardiasennus sisältää htaccess-tiedoston, jonka avulla Drupal rajoittaa url-osoitteen kautta pääsemisen ei haluttuihin kansioihin tai tiedostoihin. Virhekoodi 404 tilanteissa htaccess-tiedostossa määritetään Drupalin käsiteltäväksi. Seuraavana on Drupalin htaccess-tiedostossa määritetyt tiedostopäätteet, jotka palvelimen Apache-ohjelma estää suoraan url-osoitteessa suoriutumisen.

```
<FilesMatch
"\.(engine|inc|info|install|make|module|profile|test|po|sh|.*sql|theme
|tpl(\.php)?|xtmpl)$|^(\..*|Entries.*|Repository|Root|Tag|Template)$">
  Order allow,deny
</FilesMatch>

ErrorDocument 404 /index.php
```

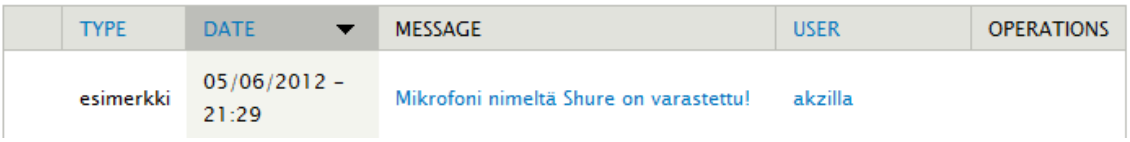
Tietokantavastaisia SQL-injektioita Drupal estää Database API -rajapinnalla. Database API tuo standardin Drupaliin, jonka avulla Drupal kehittäjät voivat luoda tietokantaa käsitteleviä toimintoja tietoturvallisesti. Tarkemmin Database API -rajapinnasta opinnäytetyön moduulien koostumuksessa. (DCD 2011c)

Drupal mahdollistaa https-protokollan käytön. Https-protokollan käyttäminen muodostaa palveluntarjoajalle kustannuksia. Palveluntarjoaja mahdollisesti haluaa rajata https-protokollan käyttöä internetsivustolla. Drupalissa on mahdollista suojata kaikki internetsivut tai määrittää erikseen internetsivuja, jotka ovat näkyvissä vain https-protokolla. (DCD 2012)

Internetsivuston latauksen yhteydessä Drupal tarkistaa settings.php-tiedoston oikeudet. Drupal ilmoittaa Drupaliin kirjautuneelle ylläpitohenkilölle automaattisesti settings.php-tiedoston oikeuksista. Suositus oikeustila tiedostoille on 444 eli vain lukuoikeus. Tätä tietoturvatoimenpidettä kutsutaan pienimmän oikeuden periaatteeksi, joka on englanniksi principle of least privilege. (Pienimmän oikeuden periaate 2011, Principle of least privilege 2011)

Drupal sisältää tapahtumaraportointiominaisuuden. Drupal-raportointiin kirjautuu ylläpitotoimintojen muutokset. Drupal tarjoaa watchdog-funktion moduulien kehittäjille. Watchdog-funktiolla ilmoitetaan Drupal-raportoinnille moduulin nimi ja raportointiviesti. Kuva 5 havainnollistaa Drupal-raportointisivulla näkyvää tapahtumailmoitusta.

```
$mikrofoni = 'Shure';
watchdog('esimerkki',
         'Mikrofoni nimeltä @mikrofoni on varastettu!',
         array('@mikrofoni' => $mikrofoni));
```



	TYPE	DATE	MESSAGE	USER	OPERATIONS
	esimerkki	05/06/2012 - 21:29	Mikrofoni nimeltä Shure on varastettu!	akzilla	

Kuva 5. Watchdog-funktion esimerkki.

IP-osotteiden estäminen onnistuu Drupalissa kahdella tavalla. Ylläpidon asetukset sivulla on mahdollista määrittää estettävä IP-osoite. Ylläpidon kautta estetty IP-osoite tallentuu tietokantaan. IP-osoitte on mahdollista estää myös settings.php-tiedostossa. Estettävä IP-osoite määritetään conf['blocked_ips']-muuttujaan taulukkomaisesti. Drupal tarkistaa ensisijaisesti settings.php-tiedostosta ennen muiden internetsivun prosessien suoritusta.

```
$conf['blocked_ips'] = array(
  '192.168.0.1',
  '192.168.255.255'
);
```

Rekisteröityneen Drupal käyttäjän salasanan salauksen vahvuus on vahva. Drupal 7 käyttää salasanan salaukseen SHA-512 hash -algoritmia. Drupal 6 käyttää md5 hash -algoritmia, joka on nykykoneille (vuonna 2012) helppo murrettavaksi. Drupal muodostaa 16 bittisen salt-merkkijonon, jossa on 2^16 = 65 536 vaihtoehtoa. Drupal 7 salasana-algoritmin kierrosten lukumäärä on 2^14 + 1 = 16 385. Salasanan merkkijono sisältää hakasulkumerkkien sisällä olevat merkit [./0-9A-Za-z]. Drupal merkitsee salasanan merkkijonon alkuun salauksen vahvuuden, S tarkoittaa SHA-512 hash-algoritmia, jonka jälkeen tulee iso C-kirjain. Seuraavana on Drupalin muodostama salattu salasanamerkkijono, joka on talletettuna tietokantaan. (Cave 2011)

```
$S$CgwilRJS4VIF1.2y0R7B4qkXJ8F8SJPcuvXRKGlMWESVXMST.5n4
```

5 OPINNÄYTETYÖN TUOTOKSET

5.1. Tupas-moduuli

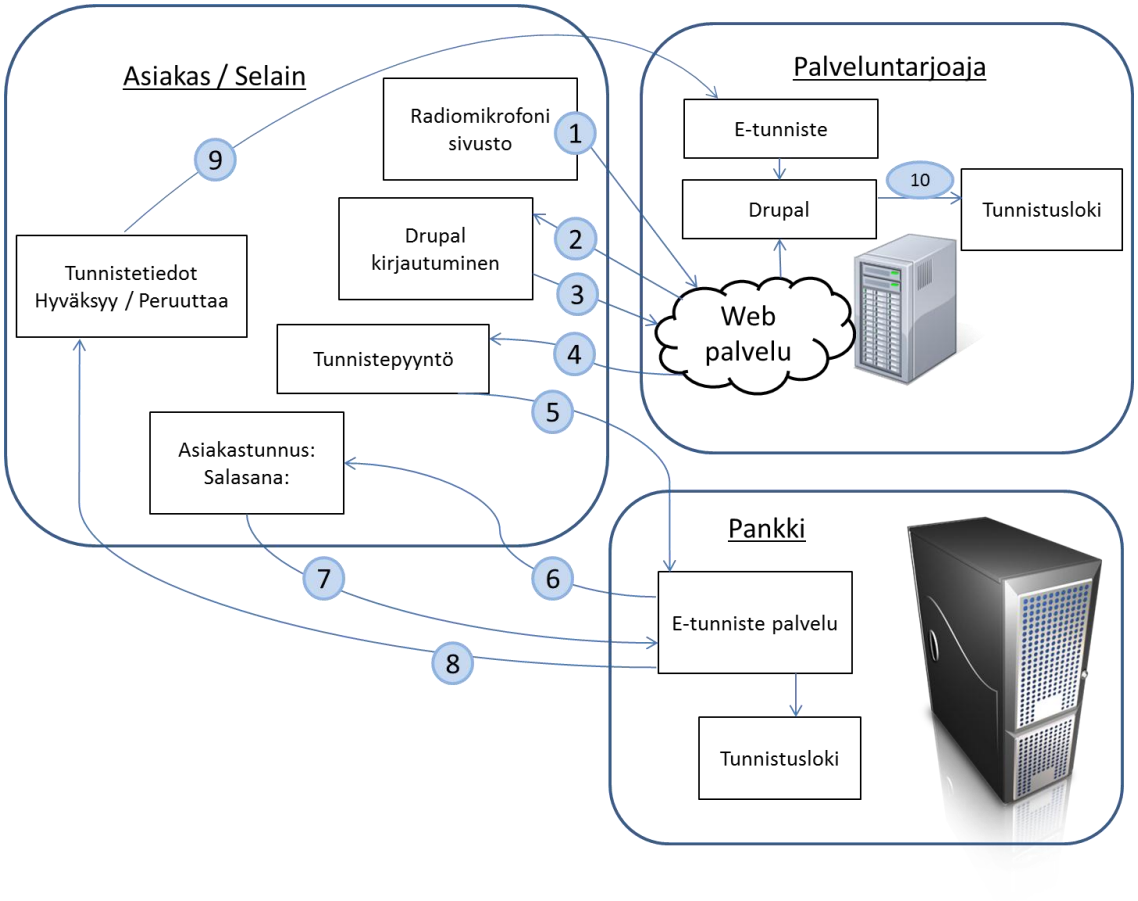
Tupas-testiversio

Ensimmäinen Tupas-testiversio suuntautui Nordean E-tunnistepalveluun. Testikoodit eivät toimi esim. Osuuspankin Tupas-palveluun. Tupas-testiversio toimii itsenäisesti php-tiedostoina. Tupas-testiversion tiedostojen ohjelma lähdeteokset ovat liitteessä 3. Saadakseen liitteen 3 mukaisen ohjelma lähdeteokset toimimaan useisiin Suomen pankkeihin, pitää html form -lomakkeen kirjainmuoto pakottaa Tupas-standardin mukaiseksi. MAC-turvatarkisteen muodostamiseksi pitää muuttaa Nordean palveluntarjoajan testitunnuksen LEHTI-merkkijonon toisen pankin ilmoittamaksi palveluntarjoajan tunnukseksi.

Hakemalla toteutettua Tupas-palvelua löysin internetistä yksinkertaisen esimerkin, jonka avulla ohjelmoin oman Tupas-testiversion. Tupas-testiversion tekemisen jälkeen havainnollistui kunnolla, miten Tupas-palvelu toimii. Testiversion jälkeen tavoitteena oli kehittää Tupas-palvelu Drupaliin toimivaksi. Ilman testiversiota olisi Drupalissa oleva Tupas-moduulin kehittäminen ollut hankalaa, koska ei olisi ollut asiakohtia, joita ajatella ja suunnitella.

Tupas-palvelun kuvaus palveluntarjoajan järjestelmässä

Kuvio 4 havainnollistaa Tupas-palvelun toiminnan palveluntarjoajan radiomikrofonisivustolla. Havainnollistamalla tunnistautumisen eri vaiheet voidaan arvioida käyttäjän vaivannäkö palvelun käyttämisessä.



Kuvio 4. Tupas-palvelun kuvaus palveluntarjoajan järjestelmästä.

Kuvion 4 kohta 1. Asiakas on palvelun käytössä vaiheessa, jolloin on mahdollista tupas-tunnistautua tai edellytetään asiakkas tupas-tunnistautumaan.

Kuvion 4 kohta 2. Asiakas ohjataan Drupal-kirjautumissivulle, jos ei ole vielä kirjautuneena palveluntarjoajan internetsivustolle. Asiakkaan ollessa kirjautuneena palveluntarjoajan internetsivustolla, asiakas ohjataan suoraan tunnistepyyntösivulle (kuvio 4 kohta 4).

Kuvion 4 kohta 3. Asiakkaan Drupal-käyttäjätunnukset lähetetään palveluntarjoajan palvelimelle tarkistettavaksi. Tietojen ollessa oikein, ohjataan asiakas tunnistepyyntösivulle (kuvio 4 kohta 4).

Kuvion 4 kohta 4. Asiakkaan ollessa kirjautuneena palveluntarjoajan internetsivustolle on asiakkaalla lupa päästä tunnistepyyntösivulle. Asiakas ohjataan pal-

veluntarjoajan Drupal-kirjautumissivulle, jos asiakas ei ole kirjautuneena (kuvio 4 kohta 2).

Kuvion 4 kohta 5. Asiakas on valinnut pankin. Asiakkaan internetselain lähettää valitun pankin html form -lomakkeen arvot post-menetelmällä asiakkaan valitsemalle pankille.

Kuvion 4 kohta 6. Pankki ohjaa asiakkaan internetsivulle, jossa asiakas kirjautuu verkkopankkitunnuksilla pankin järjestelmään. Pankki ohjaa asiakkaan takaisin palveluntarjoajan internetsivustolle, jos palveluntarjoajan muodostamassa html form -lomakkeessa on havaittu virhe.

Kuvion 4 kohta 7. Asiakaan verkkopankkitunnus lähetetään pankin palvelimelle tarkistettavaksi.

Kuvion 4 kohta 8. Pankin tarkistettua asiakkaan verkkopankkitunnuksen oikeellisuuden, asiakas ohjataan tunnistetietojen tarkistamisen internetsivulle. Tunnistetietojen internetsivulla asiakkaalle näytetään tiedot, jotka lähetetään palveluntarjoajalle.

Kuvion 4 kohta 9. Asiakkaan hyväksyessä tunnistetietojen lähettämisen palveluntarjoajalle, asiakas ohjataan palveluntarjoajan tupas-tietojen url-osoitteeseen. Tässä vaiheessa asiakkaan internetselaimelle ei lähetetä tietoja.

Kuvion 4 kohta 10. Palveluntarjoajan tarkistettua pankilta saatujen tietojen oikeellisuus talletetaan tupas-tapahtuma tietokantaan. Asiakas ohjataan radiomikrofonisivulle.

Kuvion 4 kohta 11. Asiakas voi jatkaa palveluntarjoajan palveluiden käyttöä. Asiakkaan ei tarvitse uudelleen tupas-tunnistautua kuukausiin, jos asiakas ei rekisteröi uusia radiomikrofoneja palveluntarjoajan järjestelmään.

Tupas-tietokantasuunnitelma

Tietokantasuunnitelma käyttää Drupal Schema API -määritelmiä. Tietokanta on olennainen ominaisuus Tupas-moduulissa. Tietokantaan talletetaan asiakkaan tupas-tunnistuksen tapahtuma, josta on mahdollista tarkistaa esim. onko asiakas tupas-tunnistautunut. WISE-projektin ollessa kansainvälinen, on tietokantasuunnitelma englanniksi. Taulukoissa olevat selityssarakkeet ovat ilmaisuja, jotka tulevat tietokantaan tietokantataulun sarakkeen kommentiksi.

Liite 2 sisältää kuvion radiomikrofonitietokantasuunnitelmasta. Tämän opinnäytteen tietokantataulut on väritetty oranssiksi liitteen 2 tietokantasuunnitelmassa. Vihreät tietokantataulut olen suunnitellut, mutta niitä ei ole toteutettu tämän opinnäytetyön yhteydessä. Siniset tietokantataulut ovat käytössä, mutta eivät kuulu tämän opinnäytetyön aiheeseen.

Seuraavana on taulukko 9, joka sisältää tupas_banks-tietokantataulun, jonka tarkoitus on sisältää Tupas-palvelun pankkien tiedot html form -lomakkeen muodostamiseksi. Tupas-moduulin muodostaessa jokaisesta pankista html form -lomakkeen, tarvitsee pankkien tietojen olla talletettuna tietokantaan. Tupas banks -tauluun talletetaan pankkien antamat testipankkitiedot ja niiden oikeiden pankkien tiedot, joiden kanssa palveluntarjoajalla on sopimus. Tupas banks -tauluun ei tule relaatioita, mutta taulusta tulee relaatioita.

Taulukko 9. Tupas Banks -tietokantataulu.

Sarakkeen nimi	Tyyppi	Koko	Unsigned	Ei tyhjä	Pää avain	Selitys
bank_id	serial	normal	TRUE	TRUE	TRUE	Bank ID
bank_name	varchar	60		TRUE		Original bank name.
bank_safe_name	varchar	60		TRUE		Readable bank name for every device.
bank_number	varchar	3		TRUE		Bank number for NNN.
bank_url	text	tiny		TRUE		Url address for html form action.

(jatkuu)

Taulukko 9 (jatkuu)

Sarakkeen nimi	Tyyppi	Koko	Unsigned	Ei tyhjä	Pää avain	Selitys
cert_version	varchar	4		TRUE		Certification version what bank use.
rcvid	varchar	15		TRUE		Service provider ID (receiver id).
spkey	varchar	100		TRUE		Service provider key.
keyversion	varchar	4		TRUE		Key generation version.
bank_image	varchar	50		FALSE		Bank image name for button.
enabled	int	tiny	TRUE	TRUE		To see if bank is enabled or disabled.
demo	int	tiny	TRUE	TRUE		Demo bank for testing.

Taulukon 9 bank_id-sarake on pankin rivitunnus tietokannassa. Taulukon 9 bank_name-sarake on pankin itse kutsuma pankin virallinen nimi. Taulukon 9 bank_safe_name-sarake on pankin nimi, joka ei sisällä erikoismerkkejä ja on lukukelpoinen kaikille laitteille ja ohjelmille. Taulukon 9 bank_number-sarake on pankin numeraalinen tunnus, jotka löytyvät taulukosta 5. Bank_number-sarake on varchar-tyyppinen, koska PHP käsittelee query string -merkkijonon merkkeinä.

Taulukon 9 bank_url-sarake on pankin Tupas-palveluun ohjaava url-osoite. Taulukko 9 cert_version-sarake on pankin käyttämä sertifikaattinumerotunnus, jonka pitää sisältää nollaluvut. Taulukon 9 rcvid-sarake sisältää pankin antaman tunnuksen palveluntarjoajalle, joka on uniikki jokaiselle palveluntarjoajalle.

Taulukon 9 keyversion-sarake ilmaisee käytetyn avainversion MAC-turvatarkisteen laskemisessa. Taulukon 9 bank_image-sarake on pankin logotiedoston nimi, joka on html form -lomakkeen lähetyspainikkeena. Taulukon 9 enabled-sarake ilmaisee Tupas-moduulille html form -lomaketta muodostaessa, onko pankki käytössä. Taulukon 9 demo-sarake ilmaisee, onko pankintiedot testipankkitietoja.

Seuraavana on taulukko 10, joka sisältää Tupas registered -tietokantataulun saraketiedot. Tupas registered -tietokantatauluun talletetaan asiakkaan tupas-tunnistustapahtuma. Tapahtuma kirjataan asiakkaan tunnistautumisen onnistu-

misen jälkeen sekä virhetilanteessa. Tupas registered -taulua käytetään aina, kun asiakas on rekisteröimässä uutta radiomikrofonia tai muokkaamassa rekisteröityjä radiomikrofoneja.

Taulukko 10. Tupas Registered -tietokantataulu.

Sarakkeen nimi	Tyyppi	Koko	Unsig-ned	Ei tyhjä	Viiteavain	Index	Pää-avain	Selitys
tupas_id	serial	normal	TRUE	TRUE			TRUE	An ID number of Tupas validation.
uid	int	normal	TRUE	TRUE	{users_ regi-stration} bic	TRUE		User's ID num-ber.
tupas_valid	int	tiny	TRUE	TRUE				Is Tupas valid?
validati-on_date	int	normal	TRUE	TRUE				Tupas registra-tion date in UNIX time.
bank_id	int	normal	TRUE	TRUE	{tupas_bank} bank_number			Indicates to what bank user did use.
timestamp	varchar	23		TRUE				A timestamp from bank sys-tem.

Taulukon 10 tupas_id-sarake on tupas-merkinnän tunnusnumero. Taulukon 10 uid-sarake on käyttäjän id-tunnus users-tietokantataulusta, joka on yksi suhde yhteen. Taulukon 10 tupas_valid-sarake ilmaisee, onko käyttäjän tupas-tapahtuma onnistunut tai virheellinen. Taulukon 10 validation_date-sarakkeeseen on talletettu tupas-tunnistautumisen päivämäärä, jonka avulla tarkistetaan uuden tupas-tunnistautumisen tarpeellisuuden. Taulukon 10 bank-sarake ilmaisee pankin, jota asiakas on käyttänyt. Taulukon 10 timestamp-sarakkeseen talletetaan pankilta tullut aikaleima, joka on talletettuna pankin tietokantaan tupas-tunnistautumisen yhteydessä.

Tupas-moduulin tiedostot

Opinnäytteen kirjallisessa osuudessa koodien määrää on supistettu moduulien tiedostoista. Tiedostoista on otettu kirjalliseen osuuteen olennaisimpia koodeja. Toistuvat koodit on jätetty pois. Kirjallisessa osuudessa koodit on muokattu näyttämään asialliselta ja ne eivät vastaa sovelluskehitinympäristössä olevaa koodia sataprosenttisesti. Koodin testauskoodit ja kommentoidut koodit on poistettu. Liite 1 sisältää sekvenssikaavion tupas-tunnistautumisesta. Tupas-moduulin tiedostojen lähdekoodit ovat liitteessä 4.

Admin-tiedosto

Admin-tiedosto ei ole valmis ja on rajattu empiirisen työn ulkopuolelle. Suunnitelmissa on tehdä admin-tiedostosta Tupas-moduulin ylläpitoasetuksia säätävät sivut. Ylläpitosivut näkyvät Drupal Dashboard -alustan Configuration-ylläpitosivulla. Ensisijainen ylläpitosivu on määritetty tupas.info-tiedostossa, sekä tupas_menu-funktiossa. Drupal tallentaa asetukset tietokantaan, vaikka moduulissa ei ole käsitelty asetuksia koodillisesti.

```
/**
 * Määrittää Tupas-moduulin asetuslomakkeen rungon ja
 * lataa tietokannasta aikaisemmin määritetyt asetukset.
 */
function tupas_admin_settings() {

// Moduulin asetuslomakkeen määritys, joka muodostaa valintaruu-
dut tupas-hallintasivulle.
$form[''maintain_tupas'] = array(
  '#type' => 'checkboxes',
  '#title' => t('Tupas-tunnistautumisen käyttörajoitteet'),
  '#options' => array(t('Estä tupas-tunnistautumiset'),
                 t('Vain ylläpitotestaus'),
                 t('Ota demo-pankit käyttöön'),
                 t('Vain demo-pankit käytettävissä')),
```

```
  '#default_value' => variable_get('maintain_tupas', array('page')),
  '#description' => t('Roolitasoasetukset: Drupal Dashboard
                       -> People -> Permissions'),
);

// Määrittää funktion, johon lomakkeen tiedot lähetetään käsi-
teltäväksi.
$form['#submit'][] = 'tupas_admin_settings_submit';

return system_settings_form($form);

}
```

Näkymä internetselaimessa

Pankkivalintojen lähetyspainikkeet ovat näkyvä tuotos Tupas-moduulista, jonka käyttäjä näkee internetselaimessaan. Jokainen pankin kuva sisältää yksilöidyn html form -lomakkeen. Kuvan 6 pankkien valintakuvat ovat testikuvat pankkien testi Tupas-palveluun. Internetsivulla ei ole muita graafisia näkyviä html-elementtejä.

Kuva 6. Pankkivalintojen painikkeet internetsivulla.

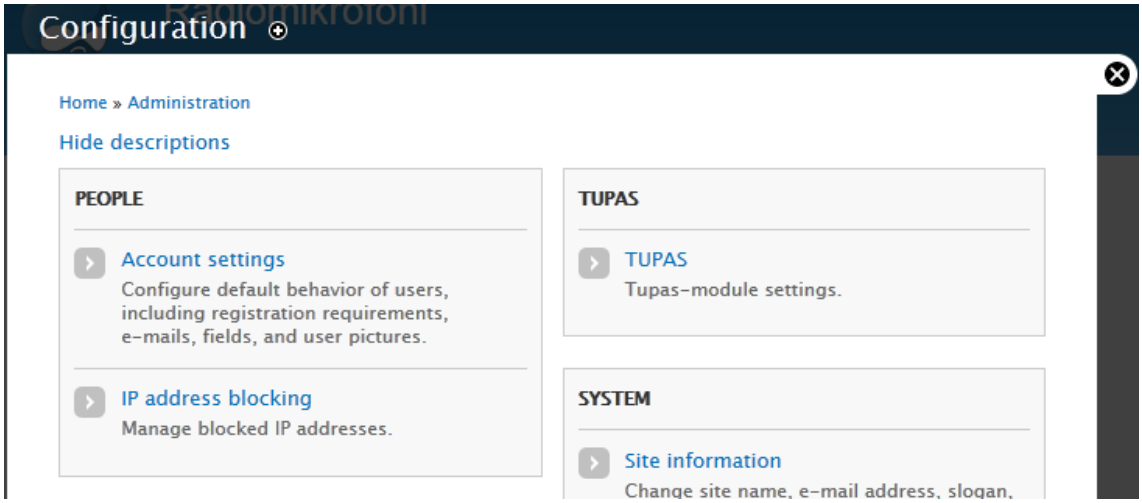
Käyttäjän painettua palveluntarjoajan internetsivuilta kuvan 6 mukaista pankki-painiketta seuraava näkymä on pankin verkkopankkitunnuksilla tunnistautumisen internetsivu. Seuraavana on kuva 7, joka havainnollistaa ensimmäisen internetsivun Nordea-pankin E-tunniste palvelussa. Kuvassa 7 käyttäjä syöttää verkkopankkitunnukset.

Kuva 7. Nordea E-tunniste pankkitunnukset.

Käyttäjän syötettyä verkkopankkitunnuksensa ja painetaan ok-nappulaa kuvan 7 mukaisella sivulla käyttäjä ohjataan tunnistetietojen yhteenvetosivulle. Seuraavana on kuva 8, joka havainnollistaa Nordea-pankin E-tunnistepalvelun tunnistetiedot internetsivun. Käyttäjä hyväksyy lähetettävät tunnistetiedot painamalla ok-painiketta. Käyttäjä ohjataan tunnistetietojen hyväksymisen jälkeen takaisin palveluntarjoajan internetsivuille.

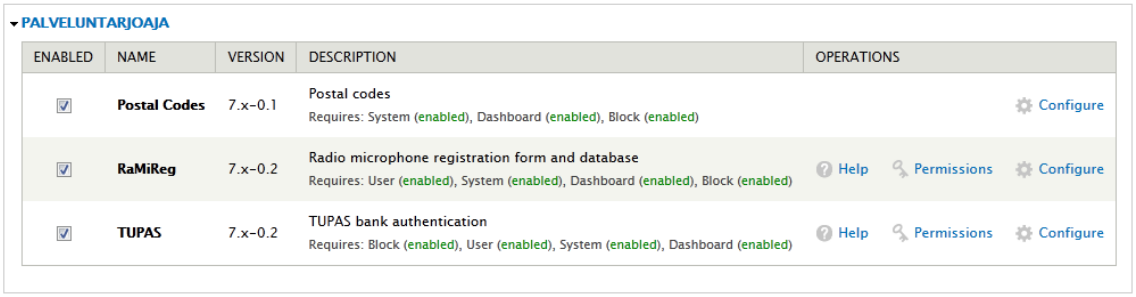
Kuva 8. Nordea E-tunniste käyttäjän tunnistetiedot

Seuraavana on kuva 9, joka havainnollistaa Drupal-ylläpitosivulla Tupas-moduulin ylläpidon valintalaatikon. Tupas-moduulin ylläpitosivulle on mahdollista päästä, myös Drupal Modules -internetsivulta, jota havainnollistaa kuva 10.



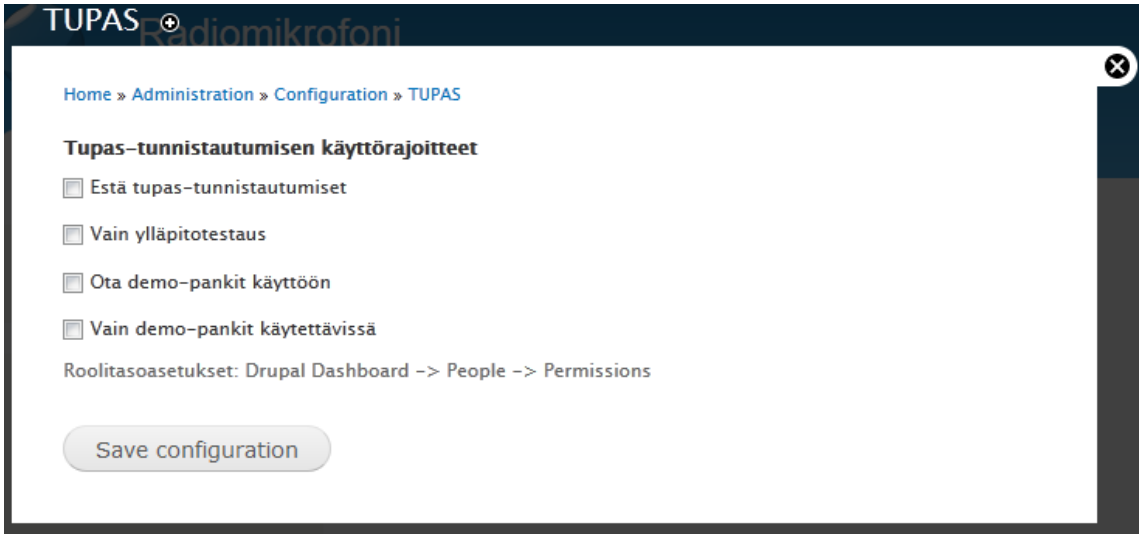
Kuva 9. Tupas-moduulin ylläpitoasetussivun valinta Drupal ylläpitosivulla.

Drupal Modules -internetsivulla on moduulin tiedot toimivuusvaatimuksista. On mahdollista siirtyä moduulin Permissions-linkin kautta asettamaan käyttäjärooli-tasorajoitteita, jos moduulilla on hook_permissions-funktio käytössä. On myös mahdollista siirtyä moduulin ylläpitosivulle. Moduulin ylläpitolinkin osoite määritetään info-tiedostossa. Kuva 10 havainnollistaa moduulien tietosivunäkymän.



Kuva 10. Drupal Modules -sivun näkymä Tupas-moduulista.

Tupas-moduulin ylläpitosivulla on mahdollista rajoittaa tai estää käyttäjien tupas-tunnistatumisen. Ylläpitosivu ei ole valmis ja sinne on tulossa uusien pankkien lisääminen tietokantaan ja pankkien käyttörajoitteita asettavat lomakkeet. Tupas-palvelulla on mahdollista pyytää useita eri tietoja asiakkaasta, joten näiden pyyntövaihtoehdot pitää tulla ylläpitosivulle mahdolliseksi. Kuva 11 havainnollistaa ylläpitosivun näkymän.



Kuva 11. Tupas-moduulin ylläpitosivu.

5.2. Postal Codes -moduuli

Tietokantasuunnitelma

Tietokantasuunnitelma käyttää Drupal Schema API -määritelmiä. Postinumeroista on olennaista tehdä tietokantaan taulu. Postinumerot tietokannassa auttavat radiomikrofoniluvan normalisointia sekä laajennetussa käyttäjätietojen normalisoinnissa. Liite 2 sisältää kuvion radiomikrofonitietokantasuunnitelmasta.

Seuraavana on taulukko 11, joka sisältää postinumerotietokantataulun rakenteen. Postinumerotietokantataulua käytetään myös AJAX hakuihin html form -lomakkeissa käyttäjän postinumeroiden löytämistä helpottamaan. Virheellisten postinumeroiden syöttäminen tietokantaan estetään postinumerotietokantataulusta tarkistamalla.

Taulukko 11. Postal Codes -tietokantataulu.

Sarakkeen nimi	Tyyppi	Koko	Ei tyhjä	Pääavain	Selitys
postal_id	serial	10	TRUE	TRUE	Postal ID in system.
postal_code	varchar	20	TRUE		Postal code
postal_name	varchar	180	TRUE		Place name
postal_country	varchar	2	TRUE		Country ISO code

Taulukon 11 postal_id-sarake on numeraalinen tunnus postinumerosta radiomikrofonitietokannassa. Post_id-sarake on pääavain, koska postinumeroja on mahdollista olla samanlaisia (Chapple 2012). Taulukon 11 Postal_code-sarake sisältää postinumeron, joka on varchar-tyyppinen. Postal_code-sarake on 20 kirjaimen pituinen, joka on GeoNames.org postinumeropalvelun ilmoittama pituus.

Taulukon 11 postal_name-sarake sisältää postitoimipaikan nimen. Postal_name-sarake on 180 kirjaimen pituinen, joka on GeoNames.org postinumeropalvelun ilmoittama pituus. Postal_country-sarake sisältää postinumeron maakohtaisen sijaintikoodin, joka on ISO-maakoodi.

Install-tiedosto

Postal codes -moduulissa ei ole muita operatiivisia tiedostoja. Postal Codes -moduulin install-tiedosto havainnollistaa Drupalin Database API tehokkuuden syötettäessä tietokantaan useita uusia rivejä. Posta Codes -moduulin lähdekoodit ovat liitteessä 5.

On kaksi tapaa tehdä postinumeroiden tietokantalisäys käyttäen Drupal Database API -rajapintaa. Ensimmäinen tapa on lisätä jokainen postinumerotieto usealla values-funktiolla tai toinen tapa on ohjelmoida esim. while-ehtolause, joka muodostaa jokaisesta uuden dp_insert-funktion.

Lisäys while-ehtolauseella, joka muodostaa tietokantalisäyksen useilla values-funktioilla suoriutuu alle kolmessa sekunnissa lisäten tietokantaan 5000 uutta riviä. While-ehtolause, joka muodostaa jokaisesta rivistä uuden db_insert-funktiolla suoriutuu 5000 uudesta rivin lisäyksestä yli 60 sekunnissa. Luvut ovat henkilökohtaisesti testattu ja en ota kantaa tietokonekohtaisia eroja.

5.3. Radiomikrofonitietokanta

Tietokantasuunnitelma käyttää Drupal Schema API -määritelmiä. WISE-projektin ollessa kansainvälinen, on tietokantasuunnitelma englanniksi. Taulukoissa olevat selityssarakkeet ovat ilmaisuja, jotka tulevat tietokantaan tietokantataulun sarakkeen selitykseksi. Tietokantasuunnitelmassa on desimaaliluvuissa käytetty numeric-datatyyppia float-datatyypin sijaan. MySQL-tietokanta tekee desimaaliluvulle pyöristyksen (Held 2009).

Liite 2 sisältää kuvion radiomikrofonitietokantasuunnitelmasta. RaMiReg Group -tietokantataulu on tarkoitettu useiden radiomikrofonien samanaikaiseen hallintaan. Asiakkaan on mahdollista hakea usealle radiomikrofonille lupa samanaikaisesti. Seuraavana on taulukko 12, joka sisältää saraketiedot RaMiReg Group -tietokantataulusta.

Taulukko 12. RaMiReg Group -tietokantataulu.

Sarakkeen nimi	Tyyppi	Koko	Unsigned	Ei tyhjä	Vakio	Viiteavain	Pääavain	Selitys
group_id	serial	normal	TRUE	TRUE			TRUE	Group ID number.
user_id	int	normal	TRUE	TRUE		{users} uid		Owner of the group.
organization_id	int	normal		FALSE		{users_ registration_ organization} organization_id		ID of user's organization.

(jatkuu)

Taulukko 12 (jatkuu).

Sarakkeen nimi	Tyyppi	Koko	Unsigned	Ei tyhjä	Vakio	Viiteavain	Pääavain	Selitys
organization_access	int	tiny	TRUE	TRUE				Indicates does other user in organization have access to this group's radio microphones.
groupname	varchar	255		TRUE				Human readable name of the group.
active	int	tiny		TRUE	1			Indicates whether radio microphone group is active.

Taulukon 12 group_id-sarake on radiomikrofoniryhmän numeraalinen tunnus ja pääavain. Taulukon 12 user_id-sarake ilmaisee radiomikrofoniryhmän omistajan. Radiomikrofoniryhmän omistaja osoittaa users-tietokantatauluun. Taulukon 12 organization_id-sarake ilmaisee radiomikrofonin esim. yrityksen tunnuksen. Organization_id-sarake osoittaa users_registeration_organization-tietokantatauluun. Taulukon 12 organization_access-sarake ilmaisee esim. yrityksen muiden käyttäjien radiomikrofoniryhmän hallintaluvan.

Taulukon 12 groupname-sarake on radiomikrofoniryhmän nimi, jonka asiakas on antanut radiomikrofoniryhmää luodessaan. Taulukon 12 active-sarake ilmaisee radiomikrofoniryhmän aktiivisuuden. Esimerkiksi radiomikrofoniryhmän radiomikrofonit ovat käytössä, joten active-sarakkeeseen asetetaan ilmaisu, että radiomikrofoniryhmä on käytössä.

Seuraavana on taulukko 13, joka sisältää RaMiReg Type -tietokantataulun saraketiedot. RaMiReg Type -tietokantataulu on pieni, mutta normalisoi tietokantaa vähentäen toistuvien tietojen tallennusta. Type-tietokantataulu sisältää radiomikrofonien tyypit. Tyyppejä ovat radiopuhelin, radiomikrofoni, radio-ohjauslaite ja telemetrialaite. Asiakkaan on mahdollista lisätä uusia radiomikrofonityyppejä.

Taulukko 13. RaMiReg Type -tietokantataulu.

Sarakkeen nimi	Tyyppi	Koko	Unsigned	Ei tyhjä	Pääavain	Selitys
type_id	serial	tiny	TRUE	TRUE	TRUE	Type ID number.
type_name	varchar	25		TRUE		Human readable type name.

Taulukon 13 type_id-sarake on radiomikrofonityypin numeraalinen tunnus ja on pääavain. Taulukon 13 type_name-sarake on radiomikrofonityypin nimitys esimerkiksi telemetrialaite.

Seuraavana on taulukko 14, joka sisältää RaMiReg Traffic type -tietokantataulun saraketiedot. RaMiReg Traffic type -tietokantataulun tarkoituksena on säilyttää radiomikrofonien liikennemuodot. Liikennemuotoja ovat dupleksi, 1-taajuus simpleksi ja 2-taajuus simpleksi.

Taulukko 14. RaMiReg Traffic type -tietokantataulu.

Sarakkeen nimi	Tyyppi	Koko	Unsigned	Ei tyhjä	Pääavain	Selitys
traffic_id	serial	tiny	TRUE	TRUE	TRUE	Traffic type ID number.
type_name	varchar	25		TRUE		Human readable type name.

Taulukon 14 traffic_id-sarake on radiomikrofonin liikennemuodon numeraalinen tunnus ja pääavain. Liikennemuotoja on vähän, joten tunnuksen kooksi on sopiva valinta tiny, joka on yksi tavu MySQL-tietokannassa. Taulukon 14 type_name-sarake on liikennemuodon nimitys, esim. dupleksi.

Seuraavana on taulukko 15, joka sisältää RaMiReg Location type -tietokantataulun saraketiedot. RaMiReg Location type -tietokantataulun on tarkoitus sisältää lähettimen eli radiomikrofonin sijaintipaikan. Lähettimen sijoitus-

paikat ovat yleensä auto, vene, kannettava. Asiakkaan on mahdollista lisätä uusia sijaintipaikkoja.

Taulukko 15. RaMiReg Location type -tietokantataulu.

Sarakkeen nimi	Tyyppi	Koko	Unsigned	Ei tyhjä	Pääavain	Selitys
location_type_id	serial	tiny	TRUE	TRUE	TRUE	Location type ID number.
locationtype_name	varchar	25		TRUE		Human readable location name.

Taulukon 15 location_type_id-sarake on liikennemuodon numeraalinen tunnus ja pääavain. Taulukon 15 locationtype_name-sarake on liikennemuodon selkokielinen nimitys, esimerkiksi auto.

Seuraavana on taulukko 16, joka sisältää RaMiReg Radio Microphone Model -tietokantataulun saraketiedot. RaMiReg Radio Microphone Model -tietokantataulu sisältää kaikki tiedossa olevat radiomikrofonilaitteiden mallit ja niiden tiedot. Taulun sisältämät tiedot ovat master dataa. Tauluun tallennetut tiedot nopeuttavat asiakkaan radiomikrofonin rekisteröimistapahtumaa vähentäen kirjoittamista. On mahdollista, että ennalta tietokantaan tallennettujen radiomikrofonilaitteiden mallitietoja ei ole mahdollista toteuttaa.

Taulukko 16. RaMiReg Radio Microphone Model -tietokantataulu.

Sarakkeen nimi	Tyyppi	Koko	Unsig-ned	Ei tyhjä	Viiteavain	Pää-avain	Selitys
rm_model_id	serial	normal	TRUE	TRUE		TRUE	Radio microphone ID.
manufacturer	varchar	100		TRUE			Manufacturer of radio microphone.
model_name	varchar	100		TRUE			Model name of radio microphone.
rm_type	int	tiny	TRUE	TRUE	{ramireg_rm_type} type_id		Indicates radio microphone’s type.
amplifier_db	numeric	20, 10		TRUE			Antenna amplifier dB count.

(jatkuu)

Taulukko 16. (jatkuu).

Sarakkeen nimi	Tyyppi	Koko	Unsig-ned	Ei tyhjä	Viiteavain	Pää-avain	Selitys
traffic_type	int	tiny	TRUE	TRUE	{ramireg_ traffictype} traffic_id		Indicates what kind of traffic does radio microphone use.
transmitter _power	numeric	20, 10		TRUE			A transmitter power in watts.
channel_width	numeric	20, 10		TRUE			How wide channel is.
more_info	varchar	255		FALSE			More information.

Taulukon 16 rm_model_id-sarake on radiomikrofonimallin numeraalinen tunnus ja pääavain. Taulukon 16 manufacturer-sarake on radiomikrofonin valmistajayrityksen nimi. Taulukon 16 model_name-sarake on radiomikrofonimallin selväkielinen nimike. Taulukon 16 rm_type-sarake ilmaisee radiomikrofonimallin tyypin. Rm_type-sarake osoittaa ramireg_rm_type-tietokantataulun type_id sarakkeeseen.

Taulukon 16 amplifier_db-sarake sisältää radiomikrofonimallin antennivahvistuksen desipelimäärän. Taulukon 16 traffic_type-sarake ilmaisee radiomikrofonimallin lähetystavan. Traffic_type-sarake osoittaa ramireg_traffictype-tietokantataulun traffic_id sarakkeeseen. Taulukon 16 transmitter_power-sarake on lähettimen teho watteina. Taulukon 16 channel_width-sarake ilmaisee radiomikrofonimallin lähetystaajuuden leveyden, esim. 25 kHz. Taulukon 16 more_info-sarake mahdollistaa radiomikrofonimallille lisätietojen tallettamisen. Lisätieto on vaihtoehtoinen ja on mahdollista olla tyhjä kenttä.

Seuraavana on taulukko 17, joka sisältää RaMiReg Registered RMs -tietokantataulun saraketiedot. RaMiReg Registered RMs -tietokantataulu on tarkoitus sisältää asiakkaiden rekisteröidyt radiomikrofonit. Asiakkaan on mahdollista ohjata rekisteröityjä radiomikrofoneja uuteen lupaan ja vaihtaa rekisteröidyn taajuuden, käyttöpaikan ja käyttöajan.

Taulukko 17. RaMiReg Registered RMs -tietokantataulu.

Sarakkeen nimi	Tyyppi	Koko	Unsigned	Ei tyhjä	Vakio	Viiteavain	Pääavain	Selitys
rm_id	serial	normal	TRUE	TRUE			TRUE	A radio microphone ID.
uid	int	normal	TRUE	TRUE		{users} uid		User's ID
organization_id	int	normal		FALSE		{users_registration_organization} organization_id		An ID number of user's organization.
rm_group	int	normal	TRUE	TRUE	0	{ramireg_rm_group} groupid		This indicates on what group does a radio microphone belongs to.
rm_model	int	normal	TRUE	TRUE		{ramireg_radiomicrophone_model} model_id		Indicates a radio microphone model.
rm_purpose	varchar	1024		TRUE				Description of a radio microphone's usage purpose.
rm_type_mark	varchar	100		TRUE				A radio microphone serial number or similar.
more_info	varchar	1024		FALSE				More information.
active	int	tiny		TRUE	1			Indicates whether a radio microphone is active.
organization_access	int	tiny	TRUE	TRUE	1			Indicates does other user in organization have access to this radio microphone.
register_timestamp	int	normal	TRUE	TRUE				A time when a radio microphone was registered.

Taulukon 17 rm_id-sarake on rekisteröidyn radiomikrofonin tunnus ja pääavain. Taulukon 17 uid-sarake on radiomikrofonin omistajan tunnus users-tietokantataulusta. Taulukon 17 organization_id-sarake on asiakkaan organisaation tunnus users_registeration_organization-tietokantataulusta. Taulukon 17 organization_access-sarake ilmaisee, onko käyttäjän organisaation käyttäjil-

lä lupa käsitellä rekisteröityä radiomikrofonia. Taulukon 17 rm_group-sarake ilmaisee radiomikrofoniryhmän, johon radiomikrofoni kuuluu. Rm_group-sarake on oletuksena nolla. Nolla on ramireg_rm_group-tietokantataulussa radiomikrofoniryhmä, joka ilmaisee radiomikrofonin kuuluvan, ei mihinkään ryhmään.

Taulukon 17 rm_model-sarake ilmaisee rekisteröidyn radiomikrofonin mallin. Rm_model-sarake osoittaa ramireg_radiomicrophone_model-tietokantataun rm_model_id-sarakkeeseen. Taulukon 17 rm_purpose-sarake sisältää rekisteröidyn radiomikrofonin tarkoituksen. Viestintävirasto haluaa tämän tiedon lupahakemukseen. Taulukon 17 rm_type_mark-sarake sisältää rekisteröidyn radiomikrofonin mallinumeron tai vastaavan merkkijonon, jonka valmistaja on ilmoittanut.

Taulukon 17 more_info-sarake on asiakkaan vapaasti annettava lisäinformaatio rekisteröidystä radiomikrofonista. Taulukon 17 active-sarake ilmaisee, onko radiomikrofoni käytössä. Register_timestamp-sarake ilmaisee radiomikrofonin rekisteröintipäivän.

Seuraavana on taulukko 18, joka sisältää RaMiReg Receiver -tietokantataulun saraketiedot. RaMiReg Receiver -tietokantataulu sisältää radiomikrofonin käyttöluvan hakemuksen yhteydessä vastaanottimen, joka ottaa radiomikrofonilähetyksen vastaan. Vastaanotin on mahdollista vastaanottaa useiden radiomikrofonien rekisteröityjä taajuuksia.

Taulukko 18. RaMiReg Receiver -tietokantataulu.

Sarakkeen nimi	Tyyppi	Koko	Unsigned	Ei tyhjä	Viiteavain	Pääavain	Selitys
receiver_id	serial	normal	TRUE	TRUE		TRUE	A receiver ID
rm_id	int	normal	TRUE	TRUE	{ramireg_registered_rm} rm_id		Indicates a radio microphone ID.
uid	int	normal	TRUE	TRUE	{users} uid		User’s ID
organization _id	int	normal		FALSE			An ID number of user's organization.

(jatkuu)

Taulukko 18 (jatkuu).

Sarakkeen nimi	Tyyppi	Koko	Unsigned	Ei tyhjä	Viiteavain	Pääavain	Selitys
organization_access	int	tiny	TRUE	TRUE			Indicates does other user in organization have access to this radio microphone license.
amplifier_db	numeric	15, 10		TRUE			Antenna amplifier dB count.
channel_width	numeric	20, 10		TRUE			How wide a channel is.

Taulukon 18 receiver_id-sarake on lähettimen tunnus ja pääavain. Taulukon 18 rm_id-sarake on rekisteröity radiomikrofoni, joka lähettää vastaanottimelle. Rm_id-sarake osoittaa ramireg_registered_rm-tietokantatauluun. Taulukon 18 uid-sarake ilmaisee lähettimen omistajan. Uid-sarake osoittaa users-tietokantatauluun.

Taulukon 18 organization_id-sarake ilmaisee organisaation, jossa käyttäjä on täyttänyt lupahakemuksen. Organization_id-sarake osoittaa users_registration_organization-tietokantatauluun. Taulukon 18 organization_access-sarake ilmaisee, onko organisaation muilla käyttäjillä lupa hallita vastaanotinta.

Taulukon 18 amplifier_db-sarake on antennivahvistuksen desibeli määrä. Amplifier_db-sarake mahdollisesti tulee poistumaan taulusta, koska radiomikrofonilaite sisältää antennivahvistuksen tiedon. Taulukon 18 channel_width-sarake on taajuuden kaistan leveys esim. 25 kHz. Channel_width-sarake mahdollisesti poistuu taulusta tai tarkoitus muuttuu, koska radiomikrofonilaite sisältää kanavan leveyden tiedon.

Seuraavana on taulukko 19, joka sisältää RaMiReg Transmit Frequency -tietokantataulun saraketiedot. RaMiReg Transmit Frequency -tietokantataulu sisältää kaikki rekisteröidyt taajuudet. Taajuustietoja käytetään vapaiden taajuuksien tietämiseen ja Wise-järjestelmän suojelemiseen.

Taulukko 19. RaMiReg Transmit Frequency -tietokantataulu.

Sarakkeen nimi	Tyyppi	Koko	Unsig-ned	Ei tyhjä	Viiteavain	Pää-avain	Selitys
transmit _frequency_id	serial	normal	TRUE	TRUE		TRUE	A transmit frequency ID.
transmit _frequency	numeric	20, 10	TRUE	TRUE			A transmit frequency.
receiver_id	int	normal	TRUE	TRUE	{ramireg_ receiver} receiver_id		Indicates a receiver ID.
location_id	int	normal	TRUE	TRUE	{ramireg_ station_location} rm_id		Indicates a station location ID.

Taulukon 19 transmit_frequency_id-sarake on lähetystaajuuden tunnus ja pääavain. Taulukon 19 transmit_frequency-sarake on lähetystaajuus, joka on kymmenen desimaalin tarkkuudella. Radiomikrofonilaite on lähetin. Taulukon 19 transmitter_id-sarake ilmaisee vastaanottimen, johon radiomikrofoni lähettää. Taulukon 19 location_id-sarake ilmaisee sijainnin, jossa taajuutta käytetään sekä päivämäärän ja kellonajan, jolloin taajuutta käytetään.

Seuraavana on taulukko 20, joka sisältää RaMiReg License -tietokantataulun saraketiedot. RaMiReg License -tietokantataulu yhdistää kaikki tiedot, joita tarvitaan radiomikrofonille luvan hakemiseen viestintävirastolta.

Taulukko 20. RaMiReg License -tietokantataulu.

Sarakkeen nimi	Tyyppi	Koko	Unsig sig-ned	Ei tyhjä	Va-kio	Viiteavain	Pää-avain	Selitys
license_id	serial	normal	TRUE	TRUE			TRUE	A radio microphone's license ID.
uid	int	normal	TRUE	TRUE		{users} uid		User's ID
organization _id	int	normal		FALSE		{users_ registration_ organization} organization _id		An ID number of user's organization.

(jatkuu)

Taulukko 20 (jatkuu).

Sarakkeen nimi	Tyyppi	Koko	Unsigned	Ei tyhjä	Vakio	Viiteavain	Pääavain	Selitys
organization_access	int	tiny	TRUE	TRUE				Indicates does other user in organization have access to this radio microphone license.
receiver_id	int	normal	TRUE	TRUE		{ramireg_receiver} receiver_id		Indicates a receiver ID.
rm_purpose	varchar	1024		TRUE				Description of a radio microphone usage purpose.
location_type	int	tiny	TRUE	TRUE		{ramireg_location_type} location_id		Indicates a location type where a radio microphone will be used.
postal_id	int	normal	TRUE	TRUE		{postal_codes} postal_id		Indicates a location where a radio microphone will be used.
station_name	varchar	100		TRUE				Description of a radio microphone station's name.
more_info	varchar	1024		FALSE				More information.
approval	int	tiny		TRUE	0			Indicates whether a radio microphone is approved.
register_timestamp	int	normal	TRUE	TRUE				A time when a radio microphone was registered.
license_end_time	int	normal	TRUE	TRUE				A time when a radio microphone’s license will end.

Taulukon 20 license_id-sarake on luvan tunnus ja pääavain. Taulukon 20 uid-sarake on luvan omistaja. Uid-sarake osoittaa users-tietokantatauluun. Taulukon 20 organization_id-sarake ilmaisee organisaation, jossa käyttäjä on täyttänyt lupahakemuksen. Organization_id-sarake osoittaa users_registration_organization-tietokantatauluun. Taulukon 20 organization_access-sarake ilmaisee, onko organisaation toisilla käyttäjillä lupa hallita lupahakemusta

Taulukon 20 receiver_id-sarake ilmaisee vastaanottimen, jota radiomikrofonilaite käyttää. Taulukon 20 rm_purpose-sarake sisältää radiomikrofonin tarkoituksen, joka on asiakkaan vapaasti kirjoittama selitys. Taulukon 20 location_type-sarake ilmaisee sijaintipaikan, jossa lähetin sijaitsee esim. auto. Location_type-sarake osoittaa ramireg_location_type-tietokantatauluun. Taulukon 20 postal_code-sarake ilmaisee sijaintitiedon, jossa radiomikrofonia käytetään ensimmäisellä käyttökerralla. Sijaintitietona käytetään postinumeroa. Postal_code-sarake osoittaa postal_codes-tietokantatauluun.

Taulukon 20 station_name-sarake ilmaisee lähetysaseman nimen. Lähetysaseman nimellä ei ole rajoituksia. Taulukon 20 more_info-sarake on lupaan ilmoitettava lisätieto- tai lisäselvennyskenttä. More-Info-sarake on vapaavalintainen ja asiakas saa kirjoittaa 1024 merkkiä pitkän tekstin. Taulukon 20 approval-sarake ilmaisee luvan hyväksynnän viranomaisilta. Oletuksena lupa on nollaluku, joka tarkoittaa, lupaa ei ole vielä hyväksytty. Taulukon 20 register_timestamp-sarake ilmaisee päivämäärän ja kellonajan, jolloin lupahakemus on rekisteröity tietokantaan. Taulukon 20 license_end_time-sarake ilmaisee aikakohdan, jolloin radiomikrofonin lisenssi päättyy.

Seuraavana on taulukko 21, joka sisältää RaMiReg Station Location -tietokantataulun saraketiedot. RaMiReg Station Location -tietokantataulu sisältää tarkemman sijainnin ja ajankohdan, jolloin on tarkoitus käyttää radiomikrofonia ja rekisteröityä taajuutta. Station Location -tietokantataulua hyödynnetään taajuuksien vapauttamiseen uusiokäyttöön kognitiivisille laitteille ja suojataan alueellisesti taajuuksia.

Taulukko 21. RaMiReg Station Location -tietokantataulu.

Sarakkeen nimi	Tyyppi	Koko	Unsigned	Ei tyhjä	Viiteavain	Pää-avain	Selitys
location_id	serial	normal	TRUE	TRUE		TRUE	A Location ID.
start_time	int	normal	TRUE	TRUE			A start time of location use.
end_time	int	normal	TRUE	TRUE			An end time of location use.
license_id	int	normal	TRUE	TRUE	{ramireg_ license} license_id		Indicates a radio microphone's license ID.
postal_id	int	normal	TRUE	FALSE	{postal _codes} postal_id		Indicates a location where a radio microphone will be used.
address	varchar	255		FALSE			An address where a radio microphone will be used.
geo_latitude	numeric	20, 15		FLASE			Latitude number.
geo_longitude	numeric	20, 15		FALSE			Longitude number.

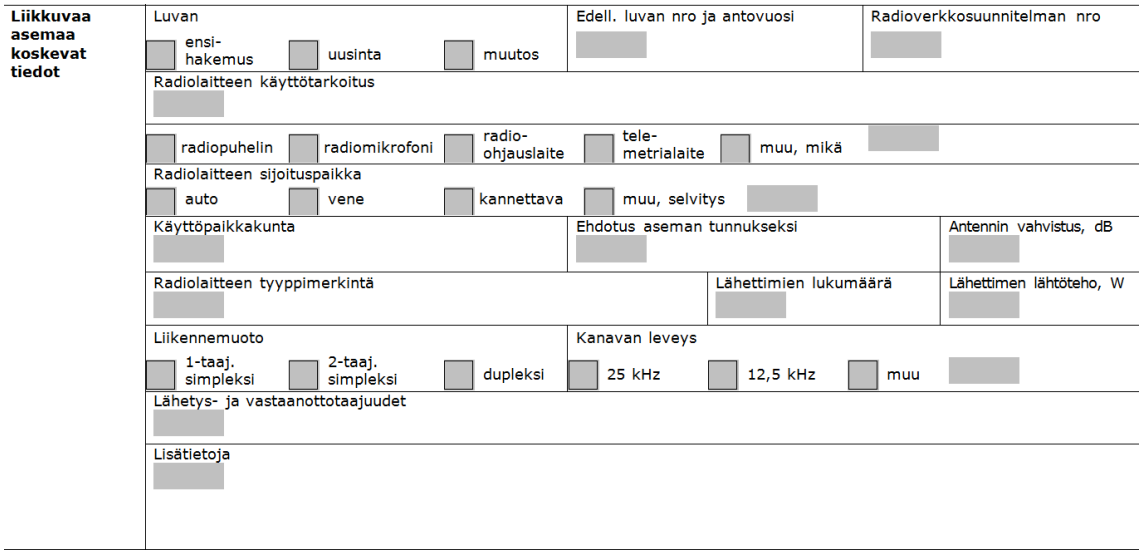
Taulukon 21 location_id-sarake on sijainnin ja ajankohdan tunnus ja pääavain. Taulukon 21 start_time-sarake ilmaisee päivämäärän ja kellonajan, jolloin taajuus otetaan käyttöön. Taulukon 21 end_time-sarake ilmaisee päivämäärän ja kellonajan, jolloin taajuus poistuu käytöstä. Taulukon 21 license_id-sarake ilmaisee lupahakemuksen, joka yhdistää lupatiedot taajuudelle ja sijainnille.

Taulukon 21 postal_id-sarake ilmaisee sijaintipaikan, jossa taajuutta käytetään. Taulukon 21 address-sarake ilmaisee osoitetiedon, jossa taajuutta käytetään. Taulukon 21 geo_latitude- ja geo_longitude-sarakkeet ilmaisevat gps-sijainnin, jossa taajuutta käytetään.

5.4. Radiomikrofonimoduuli

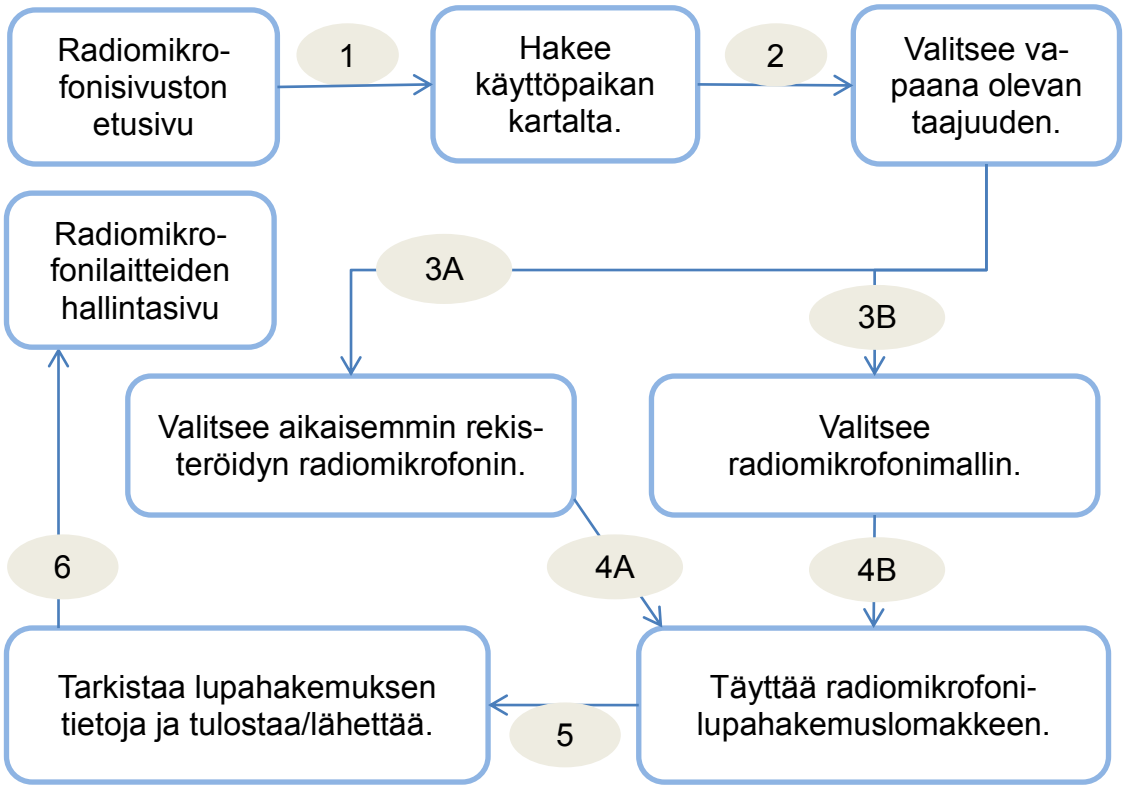
Kuvaus taajuuden- ja radiomikrofonin -rekisteröinnistä

Radiomikrofonitietokannalla on tarkoitus säilyttää radiomikrofonilaitteiden taajuuksia ja sijaintipaikkoja. Kuva 12 havainnollistaa tavan tehdä radiomikrofonille lupahakemus vuonna 2012. Radiomikrofonimoduulin on tarkoitus muuttaa kuvan 12 mukainen lupahakemuslomake sähköiseksi lupahakemukseksi. Lupahakemus lähetetään kirjeitse viestintävirastolle ja lupa myönnetään vuodeksi eteenpäin tai evätään. Tulevaisuudessa lupahakemus lähetetään internetin välityksenä viestintävirastolle. Liite 7 sisältää viestintäviraston radiomikrofonirekisteröintilomakkeen.



Kuva 12. Viestintäviraston lupahakemuslomake. (Viestintävirasto, 2011a)

Seuraavana on kuvaus suunnitelmasta, joka havainnollistaa radiomikrofonin rekisteröinnin vaiheet ja etenemisen vaihtoehdot. Etenemissuunnitelma on näkemys vaiheista, mutta saattaa muuttua tulevaisuudessa projektin jatkuessa. Käyttökokemus ja testaus mahdollisesti muuttavat etenemisen suunnitelmaa. Tavoitteena on vähentää rekisteröinnin vaiheiden määrää, jos vähentäminen on mahdollista kokemuksen ja testauksen jälkeen.



Kuvio 5. Kuvaus taajuuden- ja radiomikrofonirekisteröinnistä.

Kuvion 5 kohta 1. Asiakas etsii ja valitsee kartalta sijaintipaikan, jossa radiomikrofonia käytetään. Kartalta asiakas painaa hiirellä sijaintipaikan.

Kuvion 5 kohta 2. Asiakkaan valittua kartalta sijaintipaikan, näytetään valitun sijaintipaikan perusteella vapaat taajuudet. Kartalta asiakas näkee, myös taajuuden kuuluvuusetäisyyden.

Kuvion 5 kohta 3 vaihtoehto A. Taajuudenvalinnan jälkeen asiakas valitsee aikaisemmin rekisteröidyn radiomikrofonin. Asiakkaan on oltava kirjautuneena tästä vaiheesta eteenpäin.

Kuvion 5 kohta 3 vaihtoehto B. Asiakas valitsee hakutoiminnoilla radiomikrofonimallin. Asiakkaan on oltava kirjautuneena tai rekisteröidyttävä palveluun tästä vaiheesta eteenpäin.

Kuvion 5 kohta 4 vaihtoehto A. Asiakkaan valittua aikaisemmin rekisteröidyn radiomikrofonin, järjestelmä täyttää automaattisesti tiedot radiomikrofonilupalomakkeeseen asiakkaan puolesta.

Kuvion 5 kohta 4 vaihtoehto B. Asiakas täyttää tiedot radiomikrofonirekisteröintilomakkeen. Seuraavalla kerralla asiakkaan ei tarvitse rekisteröidä radiomikrofonia uudelleen.

Kuvion 5 kohta 5. Asiakas täyttää loput tarvittavat lupatiedot lupahakemuslomakkeesta. Lupahakemuslomake sisältää tietoja, jotka ovat muuttuvia tietoja. Esimerkiksi radiolaitteen sijoituspaikka on muuttuvaa tietoa.

Kuvion 5 kohta 6. Asiakkaan radiomikrofonilupahakemus on valmis tulostettavaksi ja lähetettäväksi viestintävirastolle. Asiakas ohjataan rekisteröityjen radiomikrofonilaitteiden hallintasivulle.

RaMiReg-moduulin tiedostot

RaMiReg-moduulin tiedostojen lähdekoodit ovat liitteessä 6.

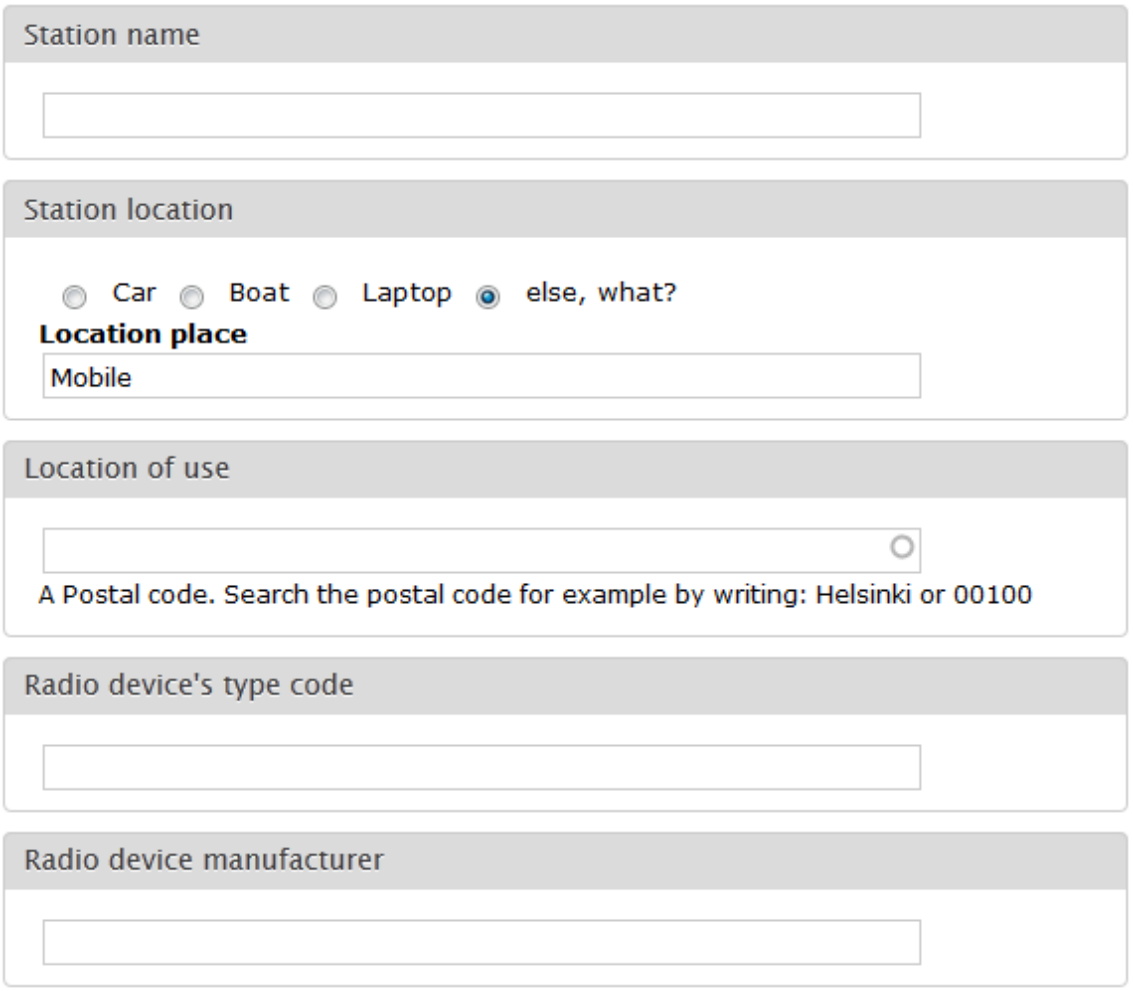
Admin-tiedosto

Admin-tiedosto on kesken ja kehityksen alla. Radiomikrofonimoduulin admin-tiedoston tarkoituksena on ohjata radiomikrofonien rekisteröintiä ylläpitosivuston kautta. Esimerkiksi estää uusien radiomikrofonien rekisteröimisen. Radiomikrofonimoduulin ylläpitosivu tulee olemaan samankaltainen kuin Tupas-moduulin ylläpitosivu. Ylläpitosivu on vielä suunnittelun alla ja uusia ideoita tulee esille radiomikrofonisivuston ensimmäisen testauksen jälkeen, jolloin kaikki sivuston ominaisuudet on yhdistettynä.

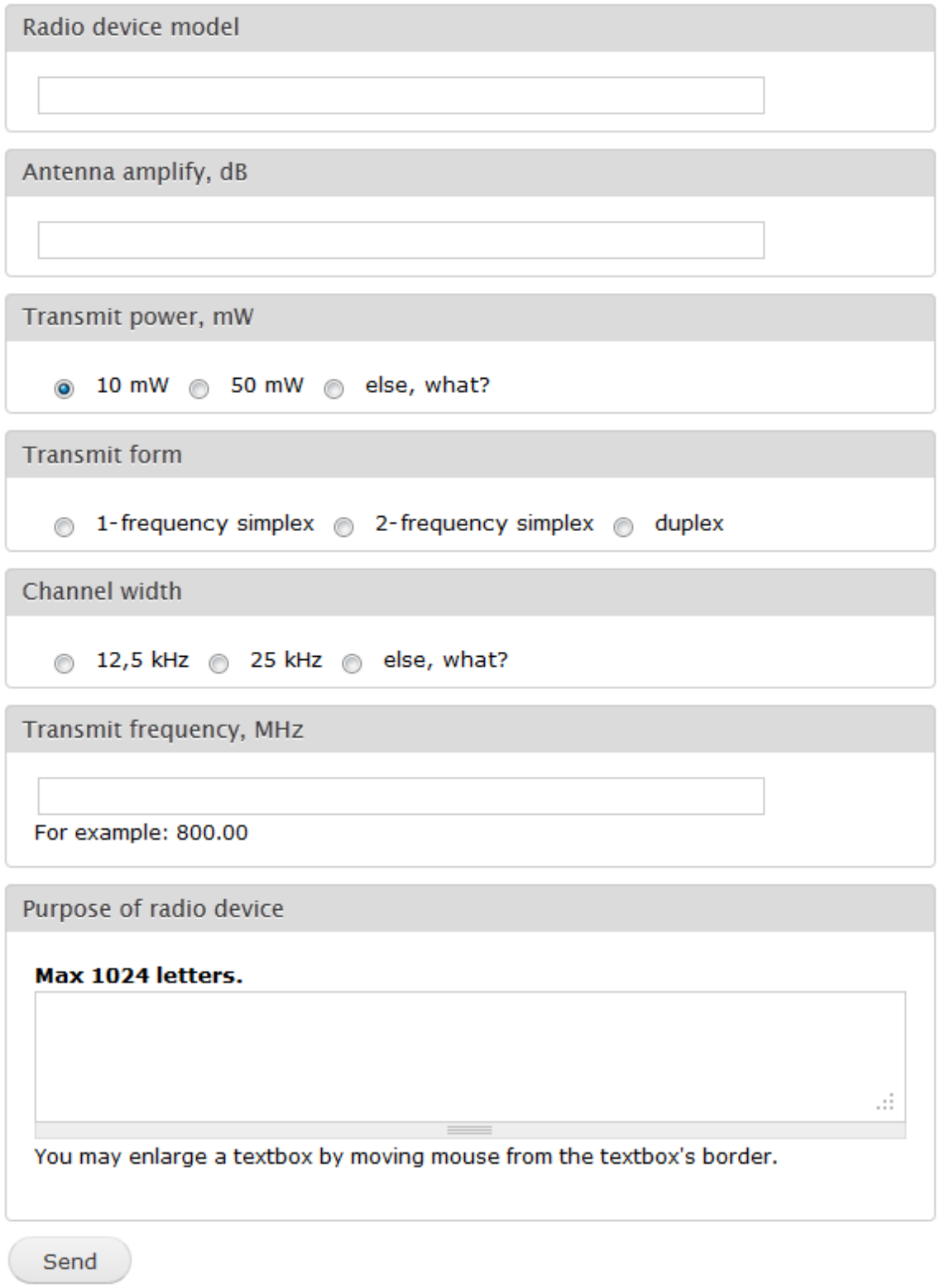
Näkymä internetselaimessa

Kuvassa 13 lomakkeen näkyvässä osassa olevaa radiolaitteen käyttötarkoitustekstilaatikkoa on mahdollista suurentaa rajattomasti, mutta sallittu maksimi pituus on 1024-merkkiä.

Käyttäjän valitessa radiolaitteen sijoituspaikan else, what? -vaihtoehdon vaihtoehtojen alle muodostuu tekstilaatikko. Tekstilaatikko poistuu näkyvistä, kun käyttäjä valitsee ylemmän vaihtoehdoista esim. Car. Postinumeroa kirjoittaessa tekstilaatikon alle muodostuu paikkavaihtoehtoja. On mahdollista hakea paikan nimellä, esim. Turku.



Kuva 13. Radiomikrofonilupahakemuslomake sähköiseksi muutettuna osa 1.



Kuva 14. Radiomikrofonilupahakemuslomake sähköiseksi muutettuna osa 2.

6 POHDINTA

Tämän opinnäytteen tekeminen alkoi mielenkiintoisesti, kun minua pyydettiin mukaan WISE-projektiin tekemään Tupas-palvelulla käyttäjän vahvaa tunnistautumista radiomikrofonirekisteröintipalveluun. Otin vastuulleni myös radiomikrofonitietokannan. Huomasin myöhemmin, että opinnäytetyöksi olisi mahdollisesti riittänyt Tupas-tunnistautuminenkin, koska töitä riitti paljon. Radiomikrofonilaitteista en tiennyt aikaisemmin mitään ja oli mielenkiintoista oppia uusia asioita.

Projektin alussa mietittiin vaihtoehtoja, miten radiomikrofonien rekisteröinti tapahtuisi. Vaihtoehtoina olivat java-sovellus ja internetsivusto. Internetsivusto valittiin, koska sen avulla kehitetään nopeammin usealle laitteelle radiomikrofonipalvelun käyttömahdollisuus. Drupal-sisällönhallintajärjestelmä valittiin, jotta ei tarvitse käyttäjien hallintaa varten koodata ja kuluttaa aikaa.

Drupal oli aivan uusi tuttavuus minulle. Oppimisessa oli kuitenkin huomattavana etuna, että Drupal on mielestäni hyvin järjestelmällinen ja sen kehitykselle on tehty standardeja. Jokainen Drupalissa oleva funktio on Drupalin API -internetsivustossa, josta on mahdollista katsoa funktion koodit ja mahdollinen sanallinen dokumentaatio. Dokumentaatiot eivät olleet aina havainnollistavia, joten itse testaamalla ja havainnollistamalla oppi parhaiten.

PHP-ohjelmointiosaaminen oli minulle eduksi projektin alussa, koska olin aloittanut PHP-ohjelmoinnin ammattikoulun aikana. PHP-ohjelmointiosaamisen soveltaminen Drupal-järjestelmään oli opettavaista, koska en ollut itse ohjelmoinut järjestelmää. Täytyi opetella Drupal kunnolla, jotta tietäisi kunnolla, mitä on tekemässä ja mitä ei pitäisi tehdä.

Radiomikrofonitietokannan suunnittelu oli opettavaista, koska se oli ensimmäinen tietokanta, jonka useiden tietokantasuunnittelukurssien ja Oracle-kurssien jälkeen tein. Tiedon keruu oli haastavaa, koska radiomikrofonilaitteet olivat uusia minulle. Kärsivällisyys ja asioiden selvittäminen oli palkitsevaa ja tuotti tulosta. Radiomikrofonitietokanta on edelleen kehityksen alla ja on mahdollista, että

pitää lisätä indeksointeja sarakkeisiin ja uusien sarakkeiden lisäyksiä tietokanta-tauluihin.

Mielestäni opinnäyte on onnistunut vaatimuksiltaan, joita henkilökohtaisesti olin asettanut. Tupas-palvelu toimii useaan pankkiin. Radiomikrofonitietokanta on MySQL-tietokantajärjestelmässä toimivana halutulla tavalla. Radiomikrofonitietokantaan on yksinkertaista lisätä uusia ominaisuuksia. Koodeissa on otettu huomioon kriittisissä paikoissa virhetilanteiden muodostumisen.

Radiomikrofonilaitteiden rekisteröinti on kehityksen alla. Projektissa seuraava vaihe on kehittää radiomikrofonilaitteiden hallintasivu, josta asiakas muuttaa esim. taajuuksia, sijainteja ja käyttöaikoja. Kehityksen alla on myös kyselyrajapinta sisään ja ulos radiomikrofonisivuston palvelimelta, jonka kautta radiomikrofonitietokannasta tiedustellaan taajuuksia ja sijainteja. Kyselyt ja tiedot käsitellään ja siirretään XML-tietona. On tiedossa, että radiomikrofonisivustolle tulee toisia vahvoja käyttäjän tunnistautumismenetelmiä esim. viranomaisten Katso-palvelu.

LÄHTEET

About Drupal 2011, Drupal Licensing. Viitattu 17.11.2011 http://drupal.org/licensing/faq.

Auger, R. 2011. Improper Output Handling. Viitattu 10.12.2011 http://projects.webappsec.org/w/page/13246934/Improper%20Output%20Handling.

Billauer, E. 2009. Why MySQL's (SQL) DATETIME can and should be avoided. Viitattu 11.12.2011 http://billauer.co.il/blog/2009/03/mysql-datetime-epoch-unix-time/.

Cave, J. 2011. Drupal 7: Secure password storage by default at last. Viitattu 11.12.2011 http://joncave.co.uk/2011/01/password-storage-in-drupal-and-wordpress/.

Chapple, M. 2012. Choosing a Primary Key. Viitattu 19.2.2012 http://databases.about.com/od/specificproducts/a/primarykey.htm.

DCD 2008. Drupal Community Documentation. Form API. Viitattu 10.12.2011 http://drupal.org/node/37775.

DCD 2011a. Drupal Community Documentation. Data types. Viitattu 17.11.2011 http://drupal.org/node/159605.

DCD 2011b. Drupal Community Documentation. Database API. Viitattu 4.12.2011 http://drupal.org/developing/api/database.

DCD 2011c. Drupal Community Documentation. Dynamic queries. Viitattu 4.12.2011 http://drupal.org/node/310075.

DCD 2012. Drupal Community Documentation. Enabling HTTP Secure (HTTPS). Viitattu 11.12.2011 http://drupal.org/https-information.

Drupal Community Documentation 2011a. Hooks. Viitattu 17.11.2011 http://api.drupal.org/api/drupal/includes--module.inc/group/hooks/7.

Drupal Community Documentation 2011b. Schema API. Viitattu 17.11.2011 http://api.drupal.org/api/drupal/includes--database--schema.inc/group/schemaapi/7.

FK 2011a. Finanssialankeskusliitto. Pankkien TUPAS-tunnistuspalvelu palveluntarjoajille. Viitattu 17.11.2011 http://www.fkl.fi/teemasivut/sahkoinen_asiointi/Dokumentit/Tupas-varmennepalvelu_v23c.pdf.

FK 2011b. Finanssialankeskusliitto. Tupas-varmennepalvelu. Viitattu 17.11.2011 http://www.fkl.fi/teemasivut/sahkoinen_asiointi/tupas/Sivut/default.aspx.

Frando 2010. Actually use foreign keys in core. Viitattu 19.2.2012 http://drupal.org/node/911352.

Held, J. 2009. MySQL Float vs. Decimal Money Datatype. Viitattu 19.2.2012 http://www.heldit.com/2009/11/database/mysql-float-vs-decimal-money-datatype/.

Liikenne- ja viestintäministeriö 2011. 800 MHz:n taajuusalueen matkaviestinkäyttö ja vapauttaminen radiomikrofonikäytöstä. Viitattu 21.2.2012 http://www.lvm.fi/c/document_library/get_file?folderId=1969043&name=DLFE-12964.doc.

Nordea Pankki Suomi Oyj 2011. E-tunniste, palvelukuvaus. Viitattu 17.11.2011 http://www.nordea.fi/sitemod/upload/Root/fi_org/liite/s/yritys/pdf/etunnist.pdf.

OP-Pohjola-ryhmä 2012. Tupas-tunnistuspalvelun käyttöehdot. Viitattu 17.11.2011 https://www.pohjola.fi/media/liitteet?cid=330256524&srcpl=4.

Paavola, J. 2011. White space test environment for broadcast frequencies (WISE). Viitattu 26.10.2011 www.tekes.fi > Ohjelmat > Trial > Aineistot > Trialin avausseminaari 7.6.2011, Helsinki.

Pienimmän oikeuden periaate 2011. Wikipedia. Viitattu 11.12.2011 http://fi.wikipedia.org/wiki/Pienimm%C3%A4n_oikeuden_periaate.

Principle of least privilege 2011. Wikipedia. Viitattu 11.12.2011 http://en.wikipedia.org/wiki/Principle_of_least_privilege.

RSnake 2011. XSS (Cross Site Scripting) Cheat Sheet. Viitattu 10.12.2011 http://ha.ckers.org/xss.html.

Salt cryptography 2011. Wikipedia. Viitattu 11.12.2011 http://en.wikipedia.org/wiki/Salt_%28cryptography%29.

Tietosuojavaltuutetun toimisto 2012. Arkaluonteisten henkilötietojen käsittely. Viitattu 16.1.2012 http://www.tietosuoja.fi/11230.htm.

Tomlinson, T. & VanDyk, J. 2010. Pro Drupal 7 Development. New York (NY):Apress, cop.

Viestintävirasto. 2011a. Liikkuvan VHF/UHF-radiolaitteen lupahakemus, LR. Viitattu 11.12.2011 http://www.viestintavirasto.fi/attachments/1156489261292/LRs.pdf.

Viestintävirasto. 2011b. Radiomikrofonit eli langattomat mikrofonit. Viitattu 19.10.2011 http://www.ficora.fi/index/luvat/luvanvaraisetradiolaitteet/radiomikrofonitelilangattomatmikrofonit.html.

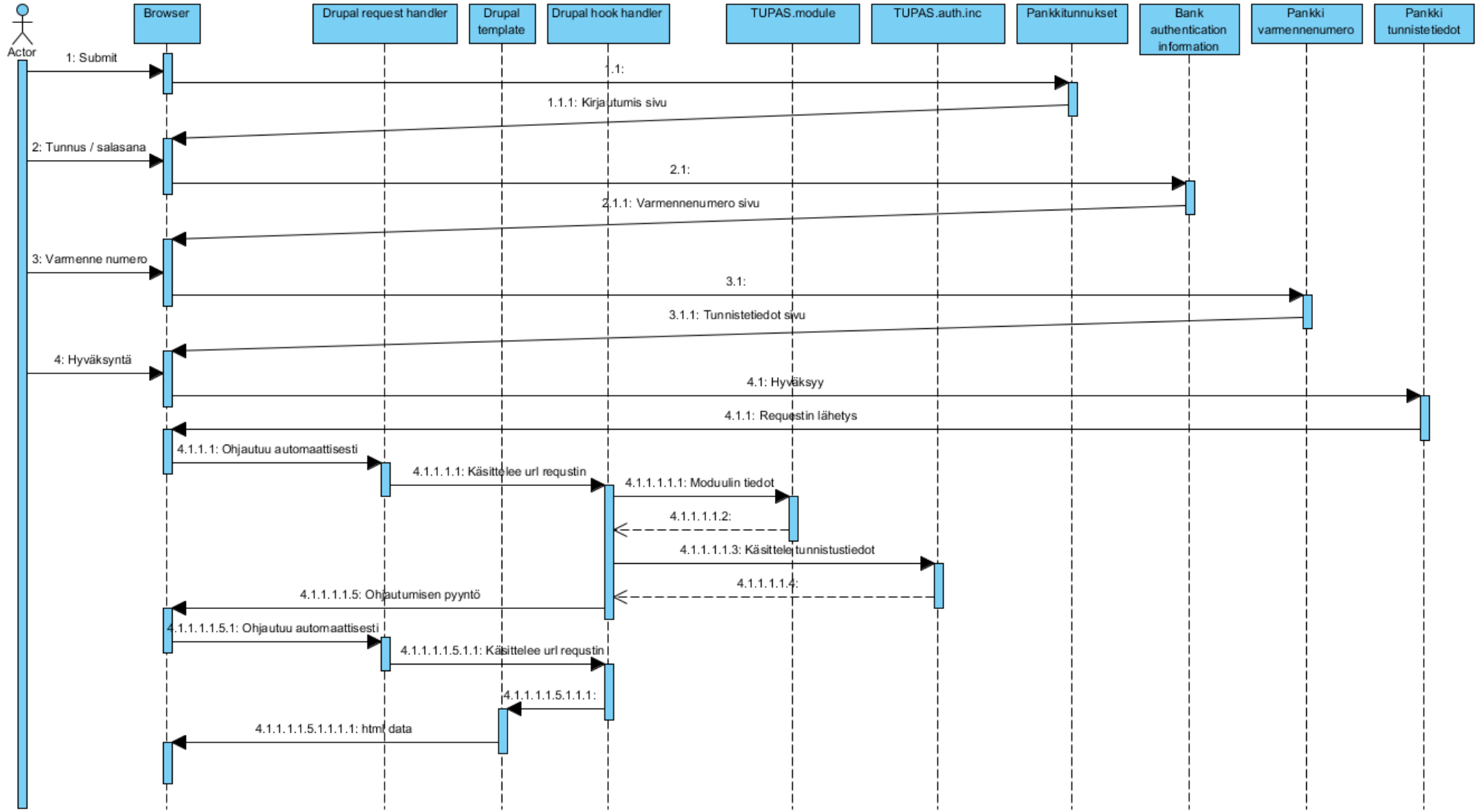
Välimäki, M. 2007. GNU Yleinen lisenssi. Viitattu 16.11.2011 http://www.turre.com/licenses/gpl_fi.html.

Webchick 2008. Remove t() from all schema descriptions. Viitattu 11.12.2011 http://drupal.org/node/332123.

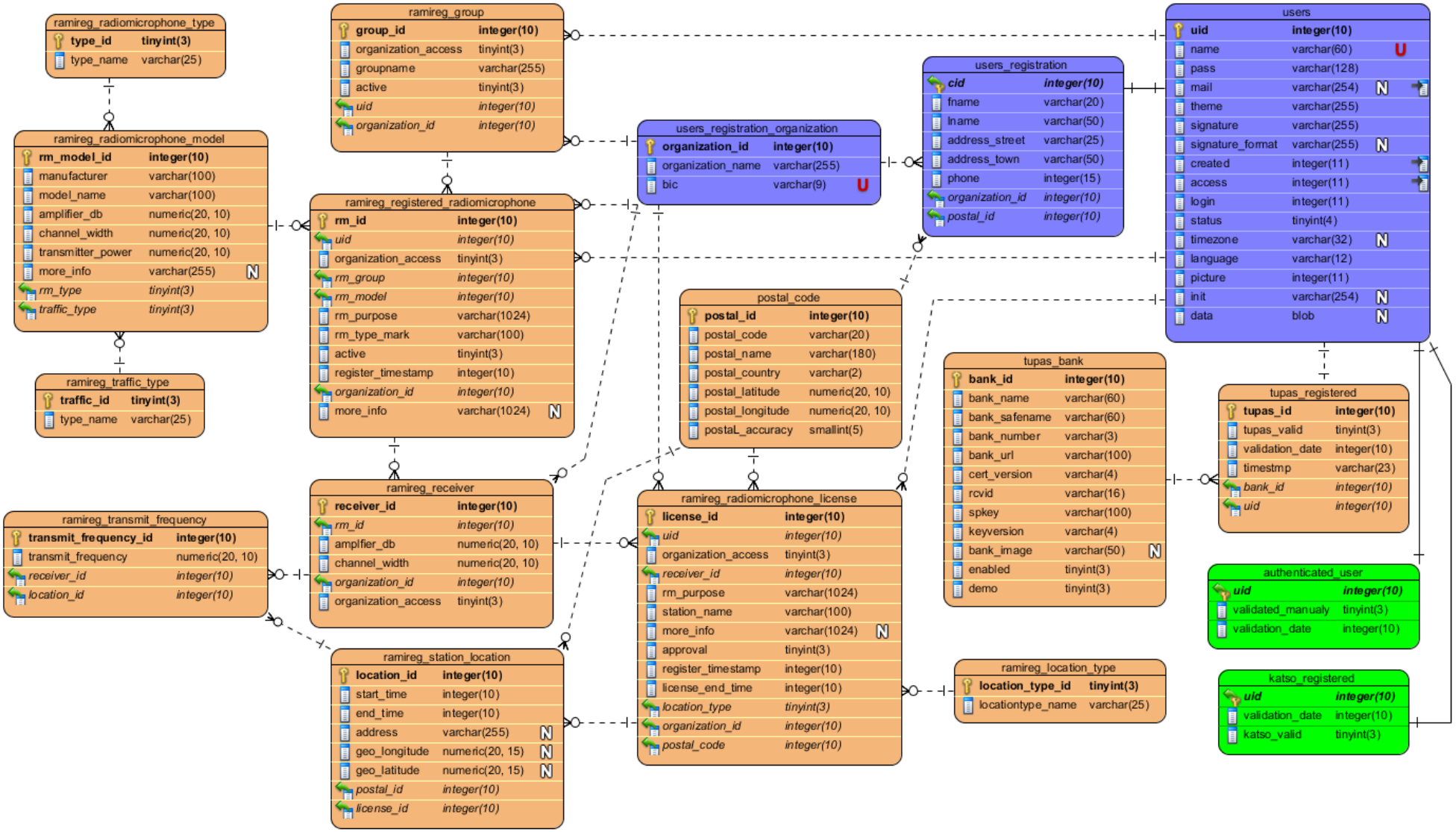
WISE. 2011a. WISE project. Viitattu 16.2.2012 wise.turkuamk.fi > Home.

WISE. 2011b. WISE overview. Viitattu 16.2.1012 wise.turkuamk.fi > Overview.

Tupas-tunnistuspyyntö sekvenssikaavio



Radiomikrofonitietokantasuunnitelma



Tupas-testiversion koodit

Pankille tupastunnistautuminen painike index.php

```
<form method="post" action="https://solo3.nordea.fi/cgi-bin/SOLO3011">

  <?php

// Määritetään lomake-taulukkomuuttujaan html form -lomakkeen elementtien
tulevat arvot.
  $lomake['ACTION_ID'] = '701';
  $lomake['VERS'] = '0002';
  $lomake['RCVID'] = '87654321';
  $lomake['LANGCODE'] = 'FI';
  $lomake['STAMP'] = date('YmdHis');
  $lomake['IDTYPE'] = '02';
  $lomake['RETLINK'] = 'http://www.esimerkki.fi/tupas/return.php';
  $lomake['CANLINK'] = 'http://www.esimerkki.fi/tupas/cancel.php';
  $lomake['REJLINK'] = 'http://www.esimerkki.fi/tupas/reject.php';
  $lomake['KEYVERS'] = '0001';
  $lomake['ALG'] = '01';

// Muodostaa html form -lomakkeen elementteihin tulevista arvoista MAC-tur-
vatarkistemerkkijonon.
  $macMerkkijono = implode('&' ,$lomake) . '&' . 'LEHTI' . '&';
  $macHash = hash("sha256", $macMerkkijono, false);
  $lomake['MAC'] = strtoupper($macHash);

  // Muodostaa html form -lomakkeen html-koodin.
  foreach($lomake as $key=>$value) {
    echo "<input type=\"hidden\"";
    echo "name=\"A01Y_$key\" value=\"$value\" />";
  }

  ?>

  <input type="submit" value="Submit" />

</form>
```

Vastaussanoman käsittely return.php

```php
<?php

// Alustetaan errors-muuttuja virhetietojen keräämiseksi.
$errors = '';

// Alustaa returnFields-taulukkomuuttujan alustaen palautettavien query
string -merkkijonon elementtien nimet.
$returnFields = array('VERS', 'TIMESTMP', 'IDNBR', 'STAMP',
'CUSTNAME', 'KEYVERS', 'ALG','CUSTID', 'CUSTTYPE', 'USERID', 'USERNAME');

// Alustaa return-taulukkomuuttujan vastaussanoman käsitellyille tiedoille.
$return = Array();

// For-looppi, jonka avulla käsitellään query string -merkkijono. B02K_ on
elementin nimen alkuosa.
for ($i = 0; $i < 9; $i++) {

// Tarkistaa onko query string -merkkijonon elementti tyhjä.
  if (isset($_GET['B02K_' . $returnFields[$i]])) {

// Asettaa return-taulukkomuuttujaan query string -merkkijonon sisällön. re-
turn-muuttujan alkion nimi on query string -merkkijonon elementin nimi ilman
B02K_:a.
    $return[$returnFields[$i]] =
    $_GET['B02K_' . $returnFields[$i]];

  } else {

// Jos query string -merkkijonon elementti on tyhjä lisätään errors-
muuttujaan virheilmoitus ja selite virheestä.
    $errors .= "Kentt&auml; {$returnFields[$i]} puuttuu.<br />";
  }
}

// Tarkistaa onko query string -merkkijonon MAC-turvatarkiste tyhjä.
if (isset($_GET['B02K_MAC'])) {

// Muodostaa MAC-muuttujaan turvatarkisteen merkkijonon.
  $MAC = strtoupper(md5(implode('&',$return).'&'.'LEHTI' .'&'));

// Verrataan muodostettua MAC-turvatarkistetta query string -merkkijonon MAC-
turvatarkisteesee.
  if ($MAC !== $_GET['B02K_MAC']) {

// Jos MAC-turvatarkiste ei täsmää, niin lisätään errors-muuttujaan selite
virhetapahtumasta.
   $errors .= 'Tarkiste ei t&auml;sm&auml;&auml;.<br />';
 }

} else {

// Lisätään errors-muuttujaan virheilmoitus, jos query string -merkkijonossa
ei ole MAC-turvatarkistetta.
```

```
  $errors .= 'MAC-tarkiste puuttuu.<br />';
}

// Ehtolause, jolla tarkistetaan onko virheitä tapahtunut. Jos virheitä ei
ole tapahtunut, niin lisätään status-muuttujaan OK-ilmoitus, jos on virheitä
ilmennyt, lisätään Virhe-ilmoitus.
$status = ($errors === '') ? 'OK' : 'Virhe';

?><?php echo "<?xml version='1.0' encoding='utf-8'?>"; ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>Tupas-testi</title>
</head>
<body>

  <h1>Tunnistus: <?php echo $status; ?> </h1>

<?php

// Tarkistetaan onko virheitä tapahtunut query string -merkkijonon tarkistuk-
sessa. Jos ei ole virheitä ilmennyt, niin näytetään pankilta palautuneet tie-
dot html taulukkoon muodostaen.
if ($errors === '') {

  echo '<p>K&auml;ytt&auml;j&auml; tunnistettu onnistuneesti</p>';
  echo '<p>Palautuneet tiedot:</p>';

  echo '<table>';

// Foreach-looppi käy läpi return-taulukkomuuttujan kaikki alkiot. Return-
muuttujan alkion nimi asetetaan key-muuttujaan ja alkion sisältö value-
muuttujaan. Jokainen query string -merkkijonon tiedot asetetaan uudelle html
taulukko rivile.
  foreach ($return as $key => $value) {

    echo '<tr><td>' . htmlentities($key) .'</td><td>'
                    . htmlentities($value) . '</td></tr>';
  }

  echo '</table>';

else {

// Jos on virheitä ilmennyt query string -merkkijonossa, niin tulostetaan
internet selaimeen virheet.
  echo '<p>K&auml;ytt&auml;j&auml;n tunnistuksessa
      oli seuraavat virheet:<br />' . $errors . '</p>';
}

?>

</body>
</html>
```

Tupas-moduulin lähdekoodit

Info-tiedosto

```
name = TUPAS ; Moduulin nimi.
description = TUPAS bank authentication ; Moduulin tarkoitus.
package = Palveluntarjoaja; Paketin nimi, johon moduuli kuuluu.
core = 7.x ; Drupal versio, jolle moduuli on tehty.
version = 7.x-0.2 ; Moduulin nykyinen versionumero.
php = 5.3 ; Moduuli on suunniteltu php 5.3:lle.
configure = admin/config/tupas/settings ; Moduulin ylläpitosivun osoite.

;Moduulit, jotka pitää olla käytössä Drupalissa.
dependencies[] = block
dependencies[] = user

; Drupal version on oltava 7.9 tai uudempi.
dependencies[] = system (>=7.9)
dependencies[] = dashboard

;Tiedostot, jotka Drupal tarvitsee tietää Tupas-moduulista.
files[] = tupas.module
files[] = tupas.install
files[] = tupas.admin.inc
```

Install-tiedosto

```
/**
 * Implementoi hook_install(). Suorittaa Tupas-moduulin
 * toimivuuteen liittyvät prosessit.
 */
function tupas_install() {

// Tupas_registered-tietokantatulua pitää muuttaa lisäämällä viitevain sarak-
keeseen uid Drupal käyttäjä tietokantatauluun nimeltä users. Schema API ei
luo viiteavain relaatioita toisiin tietokantatauluihin.
  $query = 'ALTER TABLE tupas_registered ADD FOREIGN KEY (uid)
            REFERENCES users (uid)';
  db_query($query)->execute();
```

```
// Asetetaan tupas-rekisteröinti validointi 0. Drupal users taulussa on nolla
käyttäjä, jolle tietoturvasyistä on suotavaa asettaa tupas validointi 0.
  db_insert('tupas_registeration')

// Määrittää sarakkeet, joihin tietoja asetetaan ja asettaa asetettavat syö-
tettävät tiedot.
  ->fields(array(
    'uid' => 0,
    'tupas_valid' => 0,
    'validation_date' => strtotime('now')
  ))

// Suorittaa asetetut tiedot.
  ->execute();

// Asettaa tupas_bank-tietokantatauluun testipankin tiedot.
  db_insert('tupas_bank')

// Luettelo sarakkeista, koska Tupas_bank-tietokantatauluun asetetaan useita
rivejä.
  ->fields(array(
    'bank_name', 'bank_safename', 'bank_number', 'bank_url',
    'cert_version', 'rcvid', 'spkey', 'keyvers', 'bank_image',
    'enabled', 'demo'
  ))

// Asettaa sarakkeisiin syötettävät pankkitiedot. Syötöissä käytetään for-
mat_string-funktiota, koska t-funktio tekee ennen merkkijonon tarkistusta
kielikäännöksen ja mahdollisesti muuttaa merkkejä ennalta-arvaamattomasti.
  ->values(array(
    'bank_name' => format_string('Nordea Pankki'),
    'bank_safename' => format_string('nordeapankki'),
    'bank_number' => format_string('200'),
    'bank_url' =>
      format_string('https://solo3.nordea.fi/cgi-bin/SOLO3011'),
    'cert_version' => format_string('0002'),
    'rcvid' => format_string('87654321'),
    'spkey' => format_string('LEHTI'),
    'keyvers' => format_string('0001'),
    'bank_image' => format_string('nordea_button_60x60px.gif'),
    'enabled' => 1,
    'demo' => 1,
  ))

// Suoritetaan asetetut tiedot.
  ->execute();
```

```
// Seuraavaksi luodaan uusi roolitaso vahvasti tunnistautuneille, sekä muute-
taan roolitasojen painoitusarvoa, koska uusi roolitaso on korkeammassa ase-
massa kuin administrator-roolitaso. Roolitasoilla parannetaan tietoturvaa ja
parannetaan käyttäjien hallintaa.

// Lisätään uusi roolitaso vahvasti tunnistautuneille käyttäkille. If-lause
tarkistaa onko roolitaso olemassa Drupalissa valmiina. Roolitaso haetaan roo-
litason nimellä.
  if(!user_role_load_by_name('tupas_authenticated_user')){

// Luo uuden roolitasoolion.
    $role = new stdClass();

// Määrittää roolitasolle nimen.
    $role->name = 'tupas_authenticated_user';

// Tallentaa uuden roolitason Drupaliin.
    user_role_save($role);
  } // If-ehtolause päättyy.

// Ladataan rid-muuttujaan administrator-roolitaso oliona. Roolitaso haetaan
roolitason nimellä.
  $rid = user_role_load_by_name('administrator');

// If-lause jolla tarkistetaan onko administrator-roolitaso painoitusarvol-
taan 2:na, roolitasopainotus alkaa 0:sta. Tarkistuksen tarkoitus on tarkistaa
onko administrator-roolitaso Drupal asennuksen vakiopaikassa.
  if($rid->weight == 2) {

// Jos administrator-roolitaso on Drupalin vakiopaikassa, muutetaan painoi-
tusarvo yksi ylöspäin.
    $rid->weight = $rid->weight + 1;

  } // If-ehtolause päättyy.

// Talletetaan administrator roolitaso-olio tietokantaan.
  user_role_save($rid);

// Ladataan rid-muuttujaan uusi vahvastitunnistautuneille oleva roolitaso.
Roolitaso on olio. Roolitaso haetaan roolitason nimellä.
  $rid = user_role_load_by_name('tupas_authenticated_user');

// Ladataan Drupal vakio rekisteröityneille oleva roolitaso.
  $locked = user_role_load_by_name('authenticated user');

// Muutetaan tupas-authenticated_user-roolitason painoitusarvo Drupal rekis-
teröityneiden roolitason yläpuolelle. Tällä muunnoksella varmistetaan, että
vahvastitunnistautuneiden roolitaso on korkeammassa asemassa.
  $rid->weight = $locked->weight + 1;
```

```
// Talletetaan tietokantaan tupas_authenticated_user roolitaso-olio.
  user_role_save($rid);

// Annetaan 'tupas_authenticated_user-roolitasolle oikeuden käyttää
strong_authenticated_user-roolirajoitetta.
  user_role_grant_permissions($rid->rid,
  array('access strong_authenticated_user'));

  $rid = user_role_load_by_name('administrator');

// Annetaan administrator-roolitasolle oikeuden käyttää
strong_authenticated_user-roolirajoitetta.
  user_role_grant_permissions($rid->rid,
  array('access strong_authenticated_user'));

// Asetetaan moduulin käyttöönottovaiheessa admin-roolitaso tupas-blockeille
tietoturvan parantamiseksi.
  user_role_grant_permissions($rid->rid,
   array('access valitse_pankki',
  'access tupas_return_validate', 'access tupas_auth_success'));

} // Install-funktio päättyy.
```

Module-tiedosto

```
/**
 * Implementoi hook_help(). Asettaa tupas-moduulille sivuston
 * moduulien ylläpitosivulle moduulin informaatiosivun.
 */
function tupas_help($path, $arg) {

// Ehtolause, joka tarkistaa url-osoitteen polun.
  if ($path == 'admin/help#tupas') {

// Palauttaa informaatiosivulla näkyvät tiedot. Tiedot luetaan tupas-moduulin
README.txt-tiedostosta.
    return format_string('!help', array('!help' =>
    nl2br(file_get_contents(dirname(__FILE__) ."/README.txt"))));

  } // If-ehtolause päättyy.

} // Tupas_help-funktio päättyy.
```

```
/**
 * Implementoi hook_permission(). Tämän function avulla lisätään
 * Drupaliin roolitasoihin rajoittava taso. Rajoittava taso
 * lisätään roolitasolle Drupal permissions sivulla tai
 * moduulin avulla koodattuna.
 */
function tupas_permission() {

// Palauttaa määritetyt tasorajoitteet taulukkona.
  return array(
// Määrittää vahvasti tunnistautuneille rajoitustason nimen.
    'access strong_authenticated_user' => array(

// Selvitus ja tarkoitus teksti rajoitustasolle. Teksti näkyy Drupal permis-
sions sivulla.
      'description' => t('Able to use all services.'),

// Rajoitteen otsikko Drupal permissions sivulla.
      'title' => t('Fair authenticated user'),

// Määrittää, että rajoitustaso estää pääsyn, jos roolitasolla ei ole oikeut-
ta rajoitteelle annettu.
      'restrict access' => TRUE,
    ),
  );

} // Tupas_permission-funktio päättyy. Lyhennetty versio.


/**
 * Implementoi hook_menu(). Asettaa tupas-moduulille
 * url-osoitteita, jossa moduulin funktioita suoriutuu.
 */
function tupas_menu() {

// Määrittää items-muuttujan alkioon url-osoitteen, jossa funktio suoriutuu.
  $items['tupas_return_validate'] = array(

    // Funktion nimi, joka suoritetaan.
    'page callback' => 'tupas_return_validate2',

    // Tiedoston nimi, jossa funktio sijaitsee.
    'file' => 'tupas_validate.inc',

    // Määrittää sivun suoritusrajoitteen, jossa
    // katsotaan käyttäjäroolitasojen rajoitteita.
    'access callback' => 'user_access',

    // Määrittää, mitä käyttäjäroolitasomääritystä katsotaan.
    'access arguments' => array('access tupas_return_validate'),
```

```
// Määrittää, että tupas_return_validate url-osoitetta ei tule näkyville na-
vigaatio blockiin tai mihinkään muualle, jotka hakevat url-linkkejä
hook_menu-funktiosta.
    'type' => MENU_CALLBACK,
  );

// Palauttaa asetetut url-soitteet ja määritykset suoritustavoista.
  return $items;

} // Tupas_menu-funktio

/**
 * Implementoi hook_block_info(). Määrittää blockit, jotka
 * moduuli luo. Blockit tulevat näkyville Blocks-sivulle, jossa
 * on mahdollista määrittää blockin sijainti sivulla.
 */
function tupas_block_info() {

// Alustaa blocks-muuttujan taulukkomuuttujaksi.
  $blocks = array();

// blocks-muuttujan alkioon määritetään blockin nimi.
  $blocks['list_buttons'] = array(

    // Informaatioteksti block:sta.
    'info' => t('A list of bank buttons.'),

    // Määrittää, että block:n sisällöstä ei luoda välimuistia.
    'cache' => DRUPAL_NO_CACHE,
  );
// Palauttaa tupas-moduulissa määritetyt block:t.
  return $blocks;

} // Tupas_block_info-funktio

/**
 * Implementoi hook_block_view(). Määrittää pyydetyn block:n si-
 * sällön. Oikean block:n määrittely perustuu tupas_block_view-
 * funktion parametrina block:n nimen perusteella.
 */
function tupas_block_view($block_name = '') {

// Switch ehtolause, joka valitsee funktion parametrina saadun block:n nimen
perusteella oikean block:n.
  switch ($block_name) {

// Block, joka muodostaa tupas-tunnistautumista varten käyttäjälle pankkiva-
lintapainikkeita.
    case 'list_buttons':
```

```
// Sisällyttää tupas.blocks.inc tiedoston suoritettavaksi ja käsiteltäväksi.
      include_once 'tupas.blocks.inc';

// Tietokantakysely, jossa haetaan tupas_bank-tietokantataulusta kaikkien
pankkien bank_id-tunnuksen. Ehtona pankkien tunnusten kyselyssä on, että pan-
kin on oltava käytössä. Myöhemmin tulee lisähetona valinnaiseksi testipankki-
en tarkistamiseen.
      $result =
      db_query('SELECT bank_id FROM tupas_bank WHERE enabled = 1');

// Asettaa rowCount-muuttujaan tietokantakyselyssä saatujen rivien määrän.
      $rowCount = $result->rowCount();

// Asettaa div html -elementin, jonka sisälle muodostuu pankkien valinta-
painikkeet.
      $content2 = '<div style="float: left; margin-top: 10px;">';


// For ehtolause, jonka suorituskertojen määrä perustuu tietokantakyselyssä
saatujen rivien määrään.
      for ($i=0; $i < $rowCount; $i++) {

// Lukee result-muuttujasta rivin ja tallettaa rivin bank-muuttujaan oliomuo-
dossa.
        $bank = $result->fetchObject();

// Drupal_get_form-funktio hake tupas_bank_buttons drupal form -määrittelmän.
Toisena parametrina lähetetään pankin tunnus numero, jonka perusteella oike-
anlainen drupal form määritelmä muodostuu.
        $content[$i] =
        drupal_get_form('tupas_bank_buttons', $bank->bank_id);

// Asetetaan parametrina content-muuttuja. Drupal_render-funktio muuttaa dru-
pal form -määritelmän html form -lomakkeeksi. Drupal_render-funktion para-
metrina pitää asettaa muuttuja.
        $content2 .= drupal_render($content[$i]);

      } // For-ehtolause päättyy

// Päättää div html -elementin ja lisää sen content2-muuttujaan.
      $content2 .= '</div>';

// Taulukkoon määritetään block:n aihe, joka tulee näkyville block:n otsikko-
na Drupal teemassa määritettyyn otsikkopaikkaan. Taulukkoon sisällytetään
block:n html-sisältö.
      $block = array(
        'subject' => format_string('Valitse pankki'),
        'content' => $content2,
      );
```

```
      break; // list_buttons

  } // Switch-ehtolause päättyy

// Palauttaa määritetyn block:n sisällön.
  return $block;

} // Tupas_block_view-funktio päättyy.


/**
 * Implementoi hook_form_alter(). Muuttaa Drupal form -lomakkeen
 * määritelmän sisältöä. hook_form_alter-funktiolla on mahdol-
 * lista muuttaa Drupal form muotoa, joita Drupal lisää Drupal
 * form -määritelmään automaattisesti oletuksena.
 */
function tupas_form_alter(&$form, $form_state, $form_id) {

// Switch ehtolause, joka tarkistaa oikean Drupal form -määritelmän. Ehto on
turvallisuussyistä, vaikka tarve on vain yhden Drupal form -määritelmän muut-
tamiseen.
  switch ($form_id) {

// Jos form_id-muuttuja sisältää tupas_pankkinappulat, niin halutut muutokset
tehdään.
    case 'tupas_pankkinappulat':

// Drupal lisää oletuksen automaattisesti jokaiseen Drupal form -määritelmään
form_build_id-, form_token- ja form_id html form -elementin. Edellä mainitut
ovat ylimääräisiä elementtejä ja mahdollisesti vaikuttaa tietoturvaan, koska
tupas pankkivalinta html form -elementtien sisältö lähetetään käyttäjän va-
litsemalle pankille.
      unset($form['form_build_id']);
      unset($form['form_token']);
      unset($form['form_id']);

      break;

  } // Switch

} // Tupas_form_alter-funktio päättyy.
```

Validate-tiedosto

```
/**
 * Tärkein funktio tupas-moduulissa. Funktio tarkistaa ja
 * vahvistaa pankilta saadut tiedot. Query string käsittely ja
 * tarkistus, jonka perusteella käyttäjän tupas-tunnistautuminen
 * talletetaan tietokantaan.
 */
function tupas_return_validate2() {

// Alustaa errors-muuttujan virhetietojen säilyttämistä varten.
  $errors = '';

// Alustaa reurnFields-muuttujan taulukoksi, joiden alkioiden nimet vastaavat
pankilta saadun query string -elementtien nimiä.
  $returnFields = array('VERS', 'TIMESTMP', 'IDNBR', 'STAMP', 'CUSTNAME',
'KEYVERS', 'ALG', 'CUSTID', 'CUSTTYPE', 'USERID', 'USERNAME');

// Alustaa response-muuttujan taulukkomuuttujaksi. Response-muuttujaan asete-
taan pankilta query string:stä saatujen elementtien sisältämät tiedot.
  $response = Array();

// Toistaiseksi oletuksena pankilta palautuneiden query string -elementtien
määrä on yhdeksän.
  $fields = 9;

// Kerää kaikki tiedot query string:stä returnFields-muuttujan taulukkoon.
For ehtolauseen kierrosten määrä perustuu fields-muuttujan lukuun ja tulevai-
suudessa pankilta pyydettyjen tietojen määrään.
  for ($i = 0; $i < $fields; $i++) {

// If ehtolause, joka tarkistaa, onko tarkistettavaa query string -elementtiä
olemassa. Ehtolauseessa tarkistetaan, sisältääkö jotain tarkistettava query
string -elementti.
    if (isset($_GET['B02K_'.$returnFields[$i]]) &&
       !empty($_GET['B02K_'.$returnFields[$i]])) {

// Asettaa value-muuttujaan query string -elementin sisällön, joka on for
ehtolauseen kierroksen returnFields-muuttujan alkion nimi.
      $value = $_GET['B02K_'.$returnFields[$i]];

// Response-muuttujaan asetetaan pankilta query string:ssä saadun elementin
sisällön. Sisältö määräytyy returnFields-muuttujan alkion nimestä, jonka va-
linta perustuu for ehtolauseen kierroksen i-muuttujan luvusta.
      $response[$returnFields[$i]] = $value;
```

```
// Tarkistaa onko for-ehtolause tehnyt 9 kierrosta. Jos kierros 9 toteutuu,
tarkistetaan, onko query string CUSTTYPE -elementin sisältö 08 tai 09. Jos
sisältö on 08 ta 09 muutetaan fields-muuttujan luvuksi 11, joka lisää for
ehtolauseen kierrosten määrää. Tämä toimenpide lisää, että for ehtolauseen
kierroksella luetaan query string:stä USERID- ja USERNAME-elementti.
      if ($i == 9 &&
         ($value == '08' || $value == '09')) $fields = 11;

    } else { // If B02K_

// Jos query string -elementtiä ei ole tai se on tyhjä, asetetaan elementin
nimi errors-muuttujaan.
        $errors .= $returnFields[$i] . ":";

    } // If B02K_ päättyy.

  } // For-ehtolause päättyy.

// Pankin numero B02K_TIMESTAMP query string -elementistä. B02K_TIMESTAMP-
elementin kolme ensimmäistä merkkiä ilmaisee pankin numeron josta query
string on lähetetty.
  $bank = substr($response['TIMESTMP'], 0, 3);

  try { // Tietokantavirheiden välttämiseksi.

// Tietokantakysely, jossa tupas_bank-tietokantataulusta haetaan pankin spkey
arvoa. Tietokantahaun where ehtona on, että query string B02K_TIMESTAMP -
elementistä saadun NNN numero, joka on bank-muuttujassa.
    $result = db_query('SELECT spkey FROM tupas_bank WHERE bank_number =
:bank', array(':bank' => $bank));

// Spkey-muuttujaan luetaan tietokantakyselyn tuloksesta.
  $spkey = $result->fetchObject()->spkey;

  } catch (Exeption $e) {

// Ilmoitetaan käyttäjälle, että on tapahtunut virhe tupas-tunnistautumisessa
ja kirjataan Drupal lokiin virhe tapahtuma.
    drupal_set_message('Tupas validation fail!');
    watchdog('Tupas', $e->getMessage());

// Ohjataan käyttäjä ohjataan tupas-tunnistautumisen virhesivulle. Virhesi-
vulla näytetään käyttäjälle asianmukainen ilmoitus.
    header("Location: tupas_auth_failure");

  } // Try päättyy
```

```
// If-ehtolause, joka tarkistaa errors-muuttujan sisällön. Jos virheitä ei
ole ilmaantunut funktion aikaisemmassa vaiheessa, voidaan tarkistaa query
string:stä MAC-turvatarkiste.
  if ($errors === '') {

// If-ehtolause, jolla tarkistetaan B02K_MAC query string -elementin olemas-
saolon ja ettei elementti ole tyhjä.
    if (isset($_GET['B02K_MAC']) && !empty($_GET['B02K_MAC'])) {

// Implode-funktio yhdistää response-taulukkomuuttujan alkioiden sisällöt
merkkijonoksi. Jokaisen taulukon sisältöjen eteen tulee &-merkki. Merkkijonon
loppuun lisätään tietokannasta haetun spkey-arvon, jonka molemmille puolille
asetetaan &-merkki.
      $macString = implode('&',$response).'&'. $spkey.'&';

// Hash-funktio luo macString-muuttujan merkkijonosta sha256-algoritmisen
merkkijonon, joka muutetaan isoiksi kirjaimiksi. Tupas-palvelun pankkien vä-
linen sopima hash-algoritmi on sha256, jota vuonna 2012 kaikkien pankkien
kuuluu käyttää.
      $MAC = strtoupper(hash("sha256", $macString, false));

// If-ehtolause, joka tarkistaa MAC-turvatarkisteen ja B02K_MAC query string
-elementin sisällön vastaavaisuuden. Jos vastaavuus ei täsmää, lisätään er-
rors-muuttujaan elementti, jossa on virhe.
      if ($MAC !== $_GET['B02K_MAC']) $errors .= 'B02K_MAC-match';

    } else { // If B02K_MAC

// Jos B02K_MAC query string -elementti puuttuu tai on tyhjä, lisätään er-
rors-muuttujaan elementti, jossa on virhe.
        $errors .= 'B02K_MAC';

    } // If B02K_MAC
  } // If errors

// If-ehtolause, jolla tarkistetaan, onko pankilta saatu query string
B02K_STAMP -elementin aikaleima sama, kuin tupas-tunnistautumistapahtuman
aloitus pankkien valintasivulla.
  $errors .=
  ($_SESSION['stamp'] === $response['STAMP'])? '':'B02K_STAMP';

// Users-globaalimuuttuja, joka sisältää Drupal käyttäjätietokantataulusta
tietoja. Users-muuttuja on oliomuuttuja.
  global $user;

// If-ehtolause, jolla tarkistetaan, onko virheitä tapahtunut. Käyttäjän tu-
pas-tunnistautuminen kirjataan tietokantaan onnistuneesti, jos virheitä ei
ole ilmennyt.
  if ($errors === '') {
```

```
    try { // Tietokantavirheiden välttämiseksi.

// Tietokantakysely, jossa haetaan käyttäjän aikaisempaa tupas-
tunnistautumistietoja.
      $result = db_query('SELECT tupas_id
      FROM tupas_registeration WHERE uid = :uid',
      array(':uid' => $user->uid));

// If-ehtolause, joka tarkistaa palautuiko tietokantakyselyssä yksi rivi.
Yksi rivi tarkoittaa, että käyttäjä on aikaisemmin tupas-tunnistautunut.
      if ($result->rowCount() == 1) {

// Päivitetään käyttäjän aikaisempaa tupas-tunnistautumistapahtuman kirjaus-
ta. Muutos riville on uusi aikaleima ja mahdollisesti uusi pankki, jos käyt-
täjä on käyttänyt toista pankkia aikaisemmasta tupas-tunnistautumisesta.
        db_update('tupas_registeration')
        ->fields(array(
          'tupas_valid' => 1,
          'validation_date' => strtotime('now'),
          'bank' => $bank,
        ))
        ->condition('uid', $user->uid, 'LIKE')
        ->execute();

      } else { // If rowCount

// Lisää käyttäjälle tupas_registeration-tietokantatauluun uuden tupas-
tunnistautumistapahtumamerkinnän.
        db_insert('tupas_registeration')
        ->fields(array(
          'uid' => $user->uid,
          'tupas_valid' => 1,
          'validation_date' => strtotime('now'),
          'bank' => $bank,
        ))
        ->execute();

      } // If rowCount

    } catch (Exeption $e) {

// Ilmoitetaan käyttäjälle, että on tapahtunut virhe tupas-tunnistautumisessa
ja kirjataan Drupal lokiin virhe tapahtuma.
      drupal_set_message('Tupas validation fail!');
      watchdog('Tupas', $e->getMessage());

// Ohjataan käyttäjä ohjataan tupas-tunnistautumisen virhesivulle. Virhesi-
vulla näytetään käyttäjälle asianmukainen ilmoitus.
      header("Location: tupas_auth_failure");
    } // Try päättyy
```

```
// Käyttäjä ohjataan tupas-tunnistautumisen onnistuminen sivulle.
    header("Location: tupas_auth_success");

  } else { // If errors

    try { // Tietokantavirheiden välttämiseksi.

// Tietokantakysely, jossa haetaan käyttäjän aikaisempaa tupas-
tunnistautumistietoja.
      $result = db_query('SELECT tupas_id
      FROM tupas_registeration WHERE uid = :uid',
      array(':uid' => $user->uid));

// If-ehtolause, jolla tarkistetaan, onko käyttäjä tupas-tunnistautunut ai-
kaisemmin.
      if ($result->rowCount() == 1) {

// Jos funktion aikana on havaittu virhe, joka on rekisteröity errors-
muuttujaan, pitää käyttäjän tupas-tunnistautuminen päivittää virheelliseksi.
Tulevaisuudessa, voisi lisätä ominaisuuden, jolla katsotaan, onko käyttäjän
aikaisempi tupas-tunnistautuminen ollut onnistunut. Tällöin virhetilanne ei
estäisi palvelun käyttöä, jonkin pienen ajan. Virhetilanteiden sääntöjä pitää
vielä kartoittaa lisää.
        db_update('tupas_registeration')
        ->fields(array(
          'tupas_valid' => 0,
          'validation_date' => strtotime('now'),
        ))
        ->condition('uid', $user->uid, 'LIKE')
        ->execute();

      } else { // If rowCount

// Lisätään tupas_registeration-tietokantauluun käyttäjän tupas-
tunnistautumisesta tapahtuma merkintä, jossa kirjataan tunnistautumisen epä-
onnistuneeksi.
        db_insert('tupas_registeration')
        ->fields(array(
          'uid' => $user->uid,
          'tupas_valid' => 0,
          'validation_date' => strtotime('now'),
          'bank' => $bank,
        ))
        ->execute();

      } // If rowCount päättyy

    } catch (Exeption $e) {
```

```
// Ilmoitetaan käyttäjälle, että on tapahtunut virhe tupas-tunnistautumisessa
ja kirjataan Drupal lokiin virhe tapahtuma.
        drupal_set_message('Tupas validation fail!');
        watchdog('Tupas', $e->getMessage());

// Käyttäjä ohjataan tupas-tunnistautumisen onnistuminen sivulle.
        header("Location: tupas_auth_failure");

        return;

    } // Try päättyy

// Käyttäjä ohjataan tupas-tunnistautumisen onnistuminen sivulle.
    header("Location: tupas_auth_failure");

  } // If errors päättyy.

// Jokin on mennyt vikaan funktion suorituksen aikana, jos funktio on suoriu-
tunut tähän saakka.
  header("Location: tupas_auth_failure");

} // Tupas_return_validate2-funktio päättyy.
```

Blocks-tiedosto

```
/**
 * Lataa pankkienvalinta block:n ja muodostaa siitä html-koodin.
 * Tämä funktio on tarkoitettu käytettäväksi silloin, kun
 * moduulin hook_menu-funktio määrittää sivun sisällön.
 * Block'eja ei pysty määräämään Drupal ylläpitosivun kautta,
 * jos sivu on määritetty moduulin kautta.
 */
function tupas_bank_choose() {

// Ladataan list_buttons block'i olio block_load-funktiolla.
  $block = block_load('tupas', 'list_buttons');

// Muodostaa ladatusta block:sta html-koodin.
  $block_content = _block_render_blocks(array($block));

// Muodostaa block:sta taulukon, jonka voi antaa drupal_render-funktiolle
muodostettavaksi. Sisältää tiedot block:sta, joita drupal_render-funktio
tarvitsee.
  $build = _block_get_renderable_array($block_content);

// Muodostaa selaimelle sopivan html-koodin, joka sisältää kaikki määrityk-
set, joita block:n muodostamiseen on määritetty.
  $block_rendered = drupal_render($build);

  return $block_rendered;

} // Tupas_bank_choose-funktio päättyy.

/**
 * Funktio määrittää pankin valintanappulan html form -
 * koodirungon ja käyttäjän näkymän.
 */
function tupas_bank_buttons($form, &$form_state, $bank) {

try {

// Tietokantakysely, joka hakee tupas_bank-tietokantataulusta kaikki tarvit-
tavat tiedot tupas-tunnistautumisen html form -lomakkeen luomiseksi. Tieto-
kantakyselyn ehtona on pankin numeraalinen id-tunnus, joka tulee funktion
parametrina.
$result = db_query('SELECT bank_url, cert_version, rcvid, spkey, bank_image
FROM tupas_bank
WHERE bank_id = :bank_id', array(':bank_id' => $bank));
```

```
// Alustaa form-muuttujan taulukoksi, johon määritetään html form -lomakkeen
runko.
$form = array();

// If-ehtolause, joka tarkistaa, että tietokantakyselyssä palautui pankin
tiedot. Ilman pankin tietoja, ei ole mahdollista luoda html form -lomaketta.
if($result->fetchObject() == 1) {

// Luetaan tietokantakyselyn haussa tullut rivi result-muuttujaan oliomuodos-
sa.
$result = $result->fetchObject();

// Asetetaan stamp-muuttujaan aikaleima.
$stamp = date('YmdHis');

// Asetetaan stamp-muuttujan sisällön stamp-sessiomuuttujaan. Stamp-
sessiomuuttujaa käytetään pankilta palautuneen query string -
aikaleimaelementin vertaamiseen.
$_SESSION['stamp'] = $stamp;

// Määrittää, että ennen html form -lomaketta tulee div-tagi. Tämän avulla
pankkivalintapainikkeet asettuvat järjestykseen internetselaimen näkymässä.
$form['#prefix'] = '<div style="float: left;">';

// Määrittää/pakottaa html form -lomakkeen kirjainmerkkimuodon ISO 8859-1 -
standardiksi, koska Tupas-palvelun pankkien välinen sopima. Tämä yliajaa Dru-
palin utf-8 -kirjainmerkkioletuksen.
$form['#attributes']['accept-charset'] = 'ISO 8859-1';

// Muuttaa html form -lomakkeen action-atribuutin osoittamaan pankin tupas-
palveluun. Tämä yliajaa Drupalin automaattisen action-atribuutti oletuksen.
$form['#action'] = $result->bank_url;

// Määrittää Tupas-palvelun numeraalisen tunnuksen. 701-numero on aina Tupas-
palvelussa vakio.
$form['A01Y_ACTION_ID'] = array(
  '#type' => 'hidden',
  '#value' => '701',
);

// Määrittää pankin käyttämän cert-version.
$form['A01Y_VERS'] = array(
  '#type' => 'hidden',
  '#value' => $result->cert_version,
);
```

```
// Määrittää palveluntarjoajan asiakastunnuksen, joka on pankin ja palvelun-
tarjoajan välinen sopima tunniste.
$form['A01Y_RCVID'] = array(
  '#type' => 'hidden',
  '#value' => $result->rcvid,
);

// Määrittää tupas-tunnistustapahtuman kielen, jonka perusteella pankki näyt-
tää tupas-tunnistustapahtumien sivut.
$form['A01Y_LANGCODE'] = array(
  '#type' => 'hidden',
  '#value' => 'FI',
);

// Määrittää html form -lomakkeen aikaleiman vvvvkkpphhmmssxxxxxx muodossa.
$form['A01Y_STAMP'] = array(
  '#type' => 'hidden',
  '#value' => $stamp,
);

// Määrittää tiedon, mitä pankilta pyydetään. Tulevaisuudessa tämä tulee ole-
maan dynaaminen, jos on tarvetta.
$form['A01Y_IDTYPE'] = array(
  '#type' => 'hidden',
  '#value' => '02',
);

// Määrittää url-soitteen, johon pankki ohjaa käyttäjän tupas-
tunnistautumisen onnistuessa.
$form['A01Y_RETLINK'] = array(
  '#type' => 'hidden',
  '#value' => 'http://' . $_SERVER['SERVER_NAME'] .
              url('tupas_return_validate'),
);

// Määrittää url-osoitteen, johon pankki ohjaa asiakkaan peruuttaessa tupas-
tunnistautumistapahtuman.
$form['A01Y_CANLINK'] = array(
  '#type' => 'hidden',
  '#value' => 'http://' . $_SERVER['SERVER_NAME'] .
              url('tupas_return_cancel'),
);

// Määrittää url-osoitteen, johon pankki ohjaa, jos tupas-tunnistautumisen
aikana on tapahtunut virhe.
$form['A01Y_REJLINK'] = array(
  '#type' => 'hidden',
  '#value' => 'http://' . $_SERVER['SERVER_NAME'] .
              url('tupas_reject'),
);
```

```
// Määrittää pankin avaimen version.
$form['A01Y_KEYVERS'] = array(
  '#type' => 'hidden',
  '#value' => $result->keyvers,
);

// Määrittää hash-algoritmin, jota käytetään MAC-turvatarkisteen. Numero 3
tarkoittaa SHA-256 algoritmia. Tämä on vakio uudella tupas-standardin mukaan.
$form['A01Y_ALG'] = array(
  '#type' => 'hidden',
  '#value' => '03',
);

// Asettaa macForm-muuttujaan html form -elementteihin tulevien sisällöt.
macForm-muuttujaa käytetään MAC-merkkijonon luomiseen.
$macForm['A01Y_ACTION_ID'] = $form['A01Y_ACTION_ID']['#value'];
$macForm['A01Y_VERS'] = $form['A01Y_VERS']['#value'];
$macForm['A01Y_RCVID'] = $form['A01Y_RCVID']['#value'];
$macForm['A01Y_LANGCODE'] = $form['A01Y_LANGCODE']['#value'];
$macForm['A01Y_STAMP'] = $form['A01Y_STAMP']['#value'];
$macForm['A01Y_IDTYPE'] = $form['A01Y_IDTYPE']['#value'];
$macForm['A01Y_RETLINK'] = $form['A01Y_RETLINK']['#value'];
$macForm['A01Y_CANLINK'] = $form['A01Y_CANLINK']['#value'];
$macForm['A01Y_REJLINK'] = $form['A01Y_REJLINK']['#value'];
$macForm['A01Y_KEYVERS'] = $form['A01Y_KEYVERS']['#value'];
$macForm['A01Y_ALG'] = $form['A01Y_ALG']['#value'];

// Muodostaa macForm-muuttujan arvoista merkkijonon implode-funktiolla ja
lisää tietokannasta pankin pankilta Tupas-palvelun sopimuksen yhteydessä saa-
dun yksilöidyn spkey tunnuksen. macString-muuttujan merkkijono yksilöi tupas-
tunnistautumistapahtuman.
$macString = implode('&', $macForm) .'&'. $result->spkey . '&';

// Määrittää html form -lomakkeen yksilöivän html form -elementin, johon ase-
tetaan macString-merkkijono, joka sisältää uuden merkkijonon jokaisella in-
ternetsivunlatauksen jälkeen.
$form['A01Y_MAC'] = array(
  '#type' => 'hidden',
  '#value' => strtoupper(hash("sha256", $macString, false)),
);

// ModeulPath-muuttujaan asetetaan moduulin sijaintipolun pankkien logokuvaa
varten. Tulevaisuudessa polun muodostaminen voi tapahtua dynaamisemmin Tupas-
moduulin ylläpitosivun kautta asetettujen kuvien sijainnilla.
$modulePath = drupal_get_path('module', 'tupas');
```

```
// Määrittää html form -lomakkeen painikkeelle pankin logokuvan.
$form['submit'] = array(
  '#type' => 'image_button',
  '#src' => $modulePath . '/images/' . $result->bank_image,
);

// Määrittää html form -lomakkeen jälkeen div-tagin lopetuksen.
$form['#suffix'] = '</div>';

} // If rowCount päättyy.

} catch (Exeption $e) {

// Ohjaa käyttäjän tupas_oops-sivulle, jos form-määrityksessä on tapahtunut
virhe.
  header("Location: tupas_oops");

} // Try päättyy.

// Palauttaa html form -lomakkeen määritelmän.
return $form;

} // Tupas_bank_buttons-funktio päättyy.
```

Postal Codes -moduulin lähdekoodit

Info-tiedosto

```
name = Postal Codes
description = Postal codes
package = Palveluntarjoaja
core = 7.x
php = 5.3
version = 7.x-0.1

; Module tiedosto pitää olla, koska muutoin Drupal ei suorita moduulin asen-
nusta.
files[] = postal_codes.module
files[] = postal_codes.install
```

Install-tiedosto

```
function postal_codes_install() {

  $row = 0;
  $values = array();

  try { // Tietokantavirheiden välttämiseksi.

// If-ehtolause, joka tarkistaa onko ladattavaa tiedostoa olemassa. Tiedoston
ollessa olemassa, avataan tiedosto handle-muuttujaan.
    if (($handle = fopen(dirname(__FILE__)
    . "/FI_muokattu_excell_comma.csv", "r")) !== FALSE) {

// While-ehtolause, jonka kierrosten lukumäärä on riippuvainen postinumeroi-
den lukumäärästä. Jokaisella kierroksella while-ehtolauseessa asetetaan data-
muuttujaan seuraava rivi handle-muuttujassa olevasta tiedostosta.
      while (($data = fgetcsv($handle, 0, ";")) !== FALSE) {

// Rivin kentät asetetaan data-taulukkomuuttujasta values-taulukkomuuttujaan,
joka sisältää kaikki postinumerotiedot.
        $values[$row]['postal_code'] = $data[1];
        $values[$row]['postal_code_int'] = $data[1];
        $values[$row]['postal_name'] = $data[2];
        $values[$row]['postal_country'] = $data[0];

        $values[$row]['postal_latitude'] = isset($data[9]) &&
                                           !is_null($data[9]) &&
                                           !empty($data[9]) ?  $data[9] : 0;
```

```
        $values[$row]['postal_longitude'] = isset($data[10]) &&
                                            !is_null($data[10]) &&
                                           !empty($data[10]) ? $data[10] : 0;

        $values[$row]['postal_accuracy'] = isset($data[11]) &&
                                           !is_null($data[11]) &&
                                           !empty($data[11]) ? $data[11] : 0;

        $row++;

      } // While-ehtolause päättyy.

// Sulkee avatun tiedoston.
      fclose($handle);

    } // If-ehtolause päättyy.

// Useita rivejä asettaessa tietokantaan, pitää tietokantatapahtuma aloittaa
tallettamalle db_insert-funktionkautta fields-funktio, jossa määritetään sa-
rakkeet, johon asetetaan postinumerotiedot.
    $query = db_insert('postal_codes')->fields(
    array('postal_code', 'postal_code_int', 'postal_name', 'postal_country',
          'postal_latitude', 'postal_longitude', 'postal_accuracy')
    );

// Foreach-ehtolause, joka käy läpi kaikki values-taulukkomuuttujan alkiot.
Values-muuttujan alkion taso tallettuu record-muuttujaan.
    foreach ($values as $record) {

// Lisää uuden tietokantaan lisättävän rivin, josta muodostuu yksi iso sql-
lause.
      $query->values($record);

    } // Foreach-ehtolause päättyy.

// Suorittaa tietokantatapahtuman määritelmät.
    $query->execute();

// Ilmoittaa käyttäjälle kuinka monta riviä tietokantaan lisättiin.
    drupal_set_message('Postal Codes, '. $row .' lines added');

  } catch (Exception $e) {

// Ilmoittaa käyttäjälle virheen, joka esti onnistuneen suorituksen.
    drupal_set_message('Postal Codes, Error: ' . $e->getMessage());

  } // Try päättyy

} // Postal_codes_install-funktio päättyy.
```

Radiomikrofonimoduulin lähdekoodit

Info-tiedosto

```
name = RaMiReg ; Moduulin nimi.
; Moduulin tarkoitus.
description = Radio microphone registration form and database
package = Palveluntarjoaja ; Paketti, johon moduuli kuuluu.
core = 7.x ; Drupal-versio, johon moduuli on suunnattu.
php = 5.3 ; PHP-versio, johon moduuli on suunnattu.
version = 7.x-0.2 ; Moduulin nykyinen versionumero.
configure = admin/config/ramireg/settings ; Moduulin ylläpitosivun osoite

; Moduulit, jotka pitää olla päällä Drupalissa, jotta moduuli toimii suunni-
tellusti.
dependencies[] = user
dependencies[] = system (>=7.9) ; Radiomikrofonimoduulin tarkan Drupal-
version vaatimusmääritys pitää tehdä Drupal moduulien vaatimuksiin.
dependencies[] = dashboard
dependencies[] = block

; Tiedostot, joista Drupal kuuluu tietää.
files[] = ramireg.module
files[] = ramireg.install
files[] = ramireg.admin.inc
```

Install-tiedosto

```
/**
 * Implementoi hook_schema-funktion.
 * Määrittää Radiomikrofonitietokannan.
 */
function ramireg_schema() {

// Radiomikrofoniryhmän tietokantataulun määritys.
$schema['ramireg_rm_group'] = array (

// Selitys tietokantaulun merkityksestä. Selitys tallentuu tietokantaan kom-
menttina.
  'description' => format_string('List of radio microphone
   groups to tell what group does radio microphone belong to.'),

// Radiomikrofoniryhmän tietokantataulun sarakkeiden määritys.
  'fields' => array(
```

```
// Group_id-sarake, joka on pääavain ja automaattisesti kasvava luku.
    'group_id' => array(
      'description' => format_string('Group ID number.'),
      'type' => 'serial',
      'size' => 'normal',
      'unsigned' => true,
      'not null' => true,
    ),

// User_id-sarake, joka viiteavain users-tietokantatauluun.
    'user_id' => array(
      'description' => format_string('Owner of the group.'),
      'type' => 'int',
      'size' => 'normal',
      'not null' => true,
      'unsigned' => true
    ),

// organization_id-sarake, joka rajoittaa radiomikrofoniryhmän käyttöä orga-
nisaation käyttäjille ja on viiteavain users_registration_organization-
tietokantauluun.
    'organization_id' => array(
      'description' => format_string('An ID of user\'s organization.),
      'type' => 'int',
      'size' => 'normal',
      'not null' => false,
    ),

// Organization_access-sarake ilmaisee, onko organisaation muilla käyttäjillä
lupa käyttää radiomikrofoniryhmää.
    'organization_access' => array(
      'description' => format_string('Indicates does other users
        in organization have access to this group\'s radio microphones.'),
      'type' => 'int',
      'size' => 'tiny',
      'unsigned' => true,
      'not null' => true,
      'default' => 0,
    ),

// Groupname-sarake, joka on radiomikrofoniryhmän nimi.
    'groupname' => array(
      'description' => format_string('Human readable name of the group.'),
      'type' => 'varchar',
      'length' => 255,
      'not null' => true
    ),
```

```
// Active-sarake ilmaisee, onko radiomikrofoniryhmä käytössä.
    'active' => array(
      'description' => format_string('Indicates whether radio
                                      microphone is active.'),
      'type' => 'int',
      'size' => 'tiny',
      'not null' => true,
      'default' => 1
    ),
  ),

// Määrittää group_id-sarakkeen pääavaimeksi.
  'primary key' => array(
    'group_id'
  ),

// Määrittää users_id- ja organization_id-sarakkeiden viiteavaimet. Nämä on
määritetty vaikka Schema API ei käsittele Drupal 7 versiossa.
  'foreign key' => array(
    'users_id' => array(
      'table' => 'users',
      'columns' => array('uid' => 'uid'),
    ),
    'organization_id' => array(
      'table' => 'users_ registration_organization',
      'columns' => array('organization_id' => 'organization_id'),
    ),
  ),

); // Ramireg_rm_group-tietokantaulun määritys päättyy.

  return $schema;

} // Ramireg_schema-funktion lyhennetty versio päättyy.



/**
 * Implementoi hook_install-funktion.
 * Asettaa Radiomikrofonitietokannan kaikkiin tauluihin
 * tarvittavat tiedot radiomikrofonijärjestelmän toimiakseen.
 */
function ramireg_install() {
```

```
// Tämän moduulin tietokantatauluihin pitää tehdä muutoksia, koska Schema API
ei suorita viiteavaimien määrityksia.
  $query = 'ALTER TABLE ramireg_rm_group ADD FOREIGN KEY (uid)
            REFERENCES users (uid)';
  db_query($query)->execute();

  $query =
  'ALTER TABLE ramireg_rm_group
  ADD FOREIGN KEY (organization_id)
  REFERENCES users_registration_organization (organization_id)';
  db_query($query)->execute();

// Alustaa ramireg_rm_group-tietokantataulun asettamalla ensimmäisen rivin.
Ryhmän tunnus on 0 ja sen tarkoitus on ilmaista, että radiomikrofonilla ei
ole ryhmää.
  db_insert('ramireg_rm_group')
  ->fields(array(
    'uid' => 0,
    'groupname' => format_string('Not defined'),
  ))
  ->execute();

} // Ramireg_install-funktion lyennetty versio.


/**
 * Implementoi hook_uninstall-funktion.
 * Tällä funktiolla autetaan Drupalia poistamaan
 * moduuli oikealla tavalla.
 */
function ramireg_uninstall() {

// Poistaa tietokannasta tietokantatauluja oikeassa järjestyksessä. Järjes-
tyksellä on merkitystä, koska tietokantatauluilla on relaatioita toisiinsa.
Drupal automaattinen moduulin poisto ei osaa poistaa tietokantatauluja kun-
nolla, jos tietokantatauluilla on relaatioita toisiinsa.
  db_drop_table('ramireg_transmit_frequency');
  db_drop_table('ramireg_station_location');
  db_drop_table('ramireg_radiomicrophone_license');
  db_drop_table('ramireg_receiver');
  db_drop_table('ramireg_registered_radiomicrophone');
  db_drop_table('ramireg_radiomicrophone_model');

} // Ramireg_uninstall-funktio päättyy.
```

Module-tiedosto

```
/**
 * Implementoi hook_permission-funktion.
 * Määrittää käyttäjäroolitasorajoituksia moduulin toiminnoille.
 */
function ramireg_permission() {

// Palauttaa roolitasorajoitteen määrityksen.
  return array(
    'access ramireg form' => array(
      'description' => t('View RMR form.'),
      'title' => t('Access RMR form'),
      'restrict access' => TRUE,
    ),
  );

} // Ramireg_permission-funktio päättyy.


/**
 * Määrittää radiomikrofonilupahakemuksen lomakkeen.
 * Funktio palauttaa html form -lomakkeen määritelmät taulukkona.
 */
function ramireg_radiomicrophone_registration_form() {

// Muodostaa fieldset html -elementin alustaksi aseman nimen kirjoittamitta-
miseksi.
  $form['asematunnus'] = array(
    '#type' => 'fieldset',
    '#title' => t('Station name'),
  );

// Määrittää html form -lomakkeen elementiksi tekstilaatikon.
  $form['asematunnus']['station_name'] = array(
    '#type' => 'textfield',
  );

// Muodostaa fieldset html -elementin alustaksi aseman sijaintipaikkojen va-
linnoille.
  $form['sijoituspaikka'] = array(
    '#type' => 'fieldset',
    '#title' => t('Station location'),
  );
```

```
// Asettaa fieldset-alustalle html radio -elementtejä jokaiselle sijainti-
paikkavalinnalle.
  $form['sijoituspaikka']['stationlocation'] = array(
    '#title' => t(''),
    '#type' => 'radios',
    '#default_value' => t('else, what?'),
    '#options' => drupal_map_assoc(array(
                  t('Car'),
                  t('Boat'),
                  t('Laptop'),
                  t('else, what?'))),
  );

// Lisää tekstilaatikon, jos käyttäjä valitsee aseman sijoituspaikaksi ”muu,
mikä?”. Muu, mikä? radio -painikkeen alapuolelle tulee esille tekstilaatikko.
  $form['sijoituspaikka']['stationlocationelse'] = array(
    '#type' => 'textfield',
    '#title' => t('Location place'),
    '#default_value' => t('Mobile'),
    '#states' => array(

// Määrittää näkyvyyden käyttäjän internetselaimessa näkyväksi, jos käyttäjä
on valinnut lomakkeesta muu, mikä? vaihtoehdon.
      'visible' => array(
        ':input[name="stationlocation"]' =>
         array('value' => t('else, what?')),
      ),
    ),
  );

// Muodostaa fieldset html -elementin alustaksi aseman käytön sijainnille.
  $form['kayttopaikkakunta'] = array(
    '#type' => 'fieldset',
    '#title' => t('Location of use'),
  );

// Määrittää html form -lomakkeen elementiksi tekstilaatikon. Tekstilaatik-
koon on määritetty Drupalin AJAX toiminto, joka suorittaa locati-
on_autocomplete-funktio Rm_manager-moduuliissa. Kyseinen funktio tarkistaa
käyttäjän kirjoittaman postinumeron tai paikan nimen postal_codes-
tietokantataulusta ja palauttaa 10 vaihtoehtoa käyttäjän internetselaimelle.
AJAX-toiminto suoriutuu käyttäjän jokaisella syötteen painalluksella.
  $form['kayttopaikkakunta']['locationuse'] = array(
    '#type' => 'textfield',
    '#maxlength' => 25,
    '#autocomplete_path' => 'rm_manager/location_autocomplete',
    '#description' => t('A Postal code. Search the postal code
                        for example by writing') . ': Helsinki or 00100'
  );

// Palauttaa määritetyn html form -lomakkeen.
  return $form;
```

```
} // ramireg_radiomicrophone_registration_form-funktion lyhennetty versio.


/**
 * Tarkistaa radiomikrofonihakemuslupalomakkeen html -elementtien arvot.
 * Tietoturvaa lisäävät toimenpiteet ja käyttäjän vahingossa tehtyjen
 * virheiden tarkistus. Olennaista on, että tarkistetaan onko elementti
 * tyhjä. Tyhjä elementti tuottaa turhaa koodia tietokantaan lisätessä.
 * Tämä funktio ilmoittaa käyttäjälle virheelliset kohdat.
 */
function ramireg_radiomicrophone_registration_validate($form, &$form_state) {

// Muodostetaan muuttujiin lomakkeen sisällöt eli arvot. Tämä toimenpide hel-
pottaa lomakkeen tietojen myöhempää käsittelyä.
  $antennivahvistus = $form_state['values']['antennadb'];
  $radiolaitteenmalli = $form_state['values']['model'];
  $sijoituspaikka = $form_state['values']['stationlocation'];
  $sijoituspaikka2 = $form_state['values']['stationlocationelse'];
  $asematunnus = $form_state['values']['station_name'];
  $tyyppimerkinta = $form_state['values']['merkinta'];
  $kayttotarkoitus = $form_state['values']['tekstilaatikko'];

// If-ehtolause, jolla tarkistetaan, onko aseman sijoituspaikkaa määritelty
lomakkeessa.
  if($sijoituspaikka2 !== '') {

// Jos ei ole sijoituspaikkaa valittuna, niin ilmoitetaan käyttäjälle, että
paikkaa ei ole valittuna.
  } elseif($sijoituspaikka === '') {

    form_set_error('stationlocation', t('Station location is empty.'));

// If-ehtolause, joka tarkistaa onko valittuna muu, mikä? radio painike ja
muusijoituspaikka on tyhjä. Jos muu, mikä? ei ole valittuna, niin silloin ei
ole internet selaimessa näkyvissä muusijainti tekstilaatikkoa, jolloin voi
olettaa, että käyttäjä ei ole aikonut määrittää muusijaintia. Jos tekstilaa-
tikon näkyvyttä ei tarkistettaisi, antaisi Drupali virheen ja käyttäjä ei
näkisi missä on lomakkeessa virhe.
  } elseif($sijoituspaikka === t('else, what?') && $sijoituspaikka2 === '') {

    form_set_error('stationlocationelse',
                   t('Station location else is empty.'));

  } // If-ehtolause päättyy.

// If-ehtolause, joka tarkistaa, onko määritetty asemalle nimi.
  if($asematunnus === '') {
    form_set_error('station_name',
    t('Aseman tunnus ei voi olla tyhjä.'));
  }
```

```
// If-ehtolause, joka tarkistaa onko radiomikrofonilaitteen tyyppimerkintää
kirjoitettu lomakkeeseen.
  if($tyyppimerkinta === '') {
    form_set_error('merkinta',
    t('Tyyppimerkintä ei voi olla tyhjä.'));
  }

// If-ehtolause, joka tarkistaa onko antennivahvistusta määritetty.
  if($antennivahvistus !== '') {

// If-ehtolause, joka tarkistaa onko käyttäjä syöttänyt numeraalisen arvon
lomakkeeseen.
    if(!is_numeric($antennivahvistus)) {
      form_set_error('antennadb',
      t('Antennivahvistuksen syöte pitää olla numeraalinen luku.'));
    }

// Jos antennivahvistusta ei ole lomakkeessa, niin ilmoitetaan virhe.
  } else {
    form_set_error('antennadb',
    t('Antennivahvistus ei voi olla tyhjä.'));
  }

// Tämä if-ehtolause tarkistaa, että radiolaitteen malli on syötetty lomak-
keeseen. Radiomikrofonilaitteiden valinta ei tällä tavalla tulevaisuudessa.
Tulevaisuudessa valinta tapahtuu AJAX-menetelmällä. Elementin tyhjyyden tar-
kistus tapahtuu tässä joka tapauksessa.
  if($radiolaitteenmalli !== '') {

// Jos mallia ei ole määritelty, ilmoitetaan virhe.
  } else {
    form_set_error('model', t('Radiolaitteen malli ei voi olla tyhjä.'));
  }

// If-ehtolause, joka tarkistaa radiomikrofonilaitteen käyttötarkoituksen
tekstilaatikon tyhjyyden.
  if($kayttotarkoitus === '') {
    form_set_error('tekstilaatikko',
    t('Radiolaitteen käyttötarkoitus ei voi olla tyhjä.'));

// Jos tekstilaatikossa on sisältöä suoriutuu else.
  } else {

// If-ehtolause, joka tarkistaa tekstilaatikon merkkien määrä.
    if(strlen($kayttotarkoitus) <= 1024) {

// Jos merkkejä on liikaa, ilmoitetaan virhe.
    } else {
      form_set_error('tekstilaatikko',
      t('Radiolaitteen käyttötarkoituksen maksimipituus on 1024 merkkiä.'));
```

```
    } // If-ehtolause päättyy.

  } // If-ehtolause päättyy.

} // Lomakkeen tarkistus-funktio lyhennetty versio päättyy.

/**
 * Käsittelee radiomikrofonihakemuslupalomakkeen html -elementtien arvot.
 * Prosessoi lähetetyt tiedot ja suorittaa tarvittavat tietokantatoiminnot
 * ja ohjaa käyttäjän oikealle internetsivulle.
 */
function ramireg_radiomicrophone_registration_submit($form_id, &$form_state)
{

// Otetaan tähän funktioon käyttöön user-muuttuja, joka sisältää olion, joka
on rekisteröityneen Drupal käyttäjän olio.
  global $user;

// Muodostetaan muuttujiin lomakkeen sisällöt eli arvot. Tämä toimenpide hel-
pottaa lomakkeen tietojen myöhempää käsittelyä.
  $asematunnus = $form_state['values']['station_name'];
  $sijoituspaikka = $form_state['values']['stationlocation'];
  $sijoituspaikka2 = $form_state['values']['stationlocationelse'];
  $kayttopaikka = $form_state['values']['locationuse'];
  $laitetyyppi = $form_state['values']['devicetype'];
  $laitetyyppi2 = $form_state['values']['devicetypeelse'];
  $tyyppimerkinta = $form_state['values']['merkinta'];
  $radiolaitteenvalmistaja = $form_state['values']['manufacturer'];
  $radiolaitteenmalli = $form_state['values']['model'];
  $antennivahvistus = $form_state['values']['antennadb'];
  $lahetysteho = $form_state['values']['transmitpower'];
  $lahetysteho2 = $form_state['values']['transmitpowerelse'];
  $liikennemuoto = $form_state['values']['transmitform'];
  $kanavanleveys = $form_state['values']['channelwidth'];
  $kanavanleveys2 = $form_state['values']['channelwidthelse'];
  $lahetystaajuus = $form_state['values']['transmitfrequency'];
  $kayttotarkoitus = $form_state['values']['tekstilaatikko'];

// Tarkistaa syötteen sisällön ja alustaa kanavaleveys-muuttujaan vastaavalla
numeraalisella arvolla.
  if($kanavanleveys === '12.5 kHz') {
    $kanavanleveys = 12.5;
  } elseif ($kanavanleveys === '25 kHz') {
    $kanavanleveys = 25;

// Jos käyttäjä on syöttänyt oman arvon, asetetaan käyttäjän määrittämä arvo.
  } elseif ($kanavanleveys === t('else, what?')) {
    $kanavanleveys = $kanavanleveys2;
  } // If-ehtolause päättyy.
```

```
// Tarkistaa syötteen sisällön ja alustaa lahetysteho-muuttujan vastaavalla
numeraalisella arvolla.
  if($lahetysteho === '10 mW') {
    $lahetysteho = 0.1;
  } elseif ($lahetysteho === '50 mW') {
    $lahetysteho = 0.5;

// Jos käyttäjä on syöttänyt oman arvon, asetetaan käyttäjän määrittämä arvo.
  } elseif ($lahetysteho === t('else, what?')) {

    $lahetysteho = $lahetysteho2;
  } // If-ehtolause päättyy.

// PHP-ohjelmointikielen yhden rivin ehtolause, joka tarkistaa käyttäjän mää-
rittämän laitetyypin ja kääntää laitetyypin englanninkieliseksi. Tietokannas-
sa laitetyypit ovat tallennettu englanninkieliseksi.
  $laitetyyppi = t('@laitetyyppi', array('@laitetyyppi'
             => ((!empty($laitetyyppi) || $laitetyyppi !== t('else, what?'))
             ? $laitetyyppi : (!empty($laitetyyppi2)
             ? $laitetyyppi2 : null))));

// Muuttaa laitetyypin-merkkijonon pieniksi kirjaimiksi.
  $laitetyyppi = strtolower($laitetyyppi);

// Tietokantakysely, jossa haetaan radiomikrofonilaitteen mallin pääavainta.
Tämän avulla muodostetaan viittaus malliin. Tämän tietokantakyselyn avulla
pyritään estämään duplikaattidatan ilmeentymisen tietokantataulussa.
  $radiolaitteenmalli_id =
  db_select('ramireg_radiomicrophone_model', 'rm_model');

  $radiolaitteenmalli_id->join('ramireg_radiomicrophone_type', 'rm_type');
  $radiolaitteenmalli_id->join('ramireg_traffic_type', 'traffic_type');

  $radiolaitteenmalli_id->fields('rm_model', array('rm_model_id'));

  $radiolaitteenmalli_id->condition('rm_model.manufacturer',
                          $radiolaitteenvalmistaja, 'LIKE');

  $radiolaitteenmalli_id->condition('rm_model.model_name',
                          $radiolaitteenmalli, 'LIKE');

  $radiolaitteenmalli_id->condition('rm_model.amplifier_db',
                          $antennivahvistus, 'LIKE');

  $radiolaitteenmalli_id->condition('rm_model.channel_width',
                  (!empty($kanavanleveys) ? $kanavanleveys : null), 'LIKE');

  $radiolaitteenmalli_id->condition('rm_model.transmitter_power',
                  (!empty($lahetysteho) ? $lahetysteho :  null), 'LIKE');
```

```
  $radiolaitteenmalli_id->condition('rm_type.type_name',
                          $laitetyyppi, 'LIKE');

  $radiolaitteenmalli_id->condition('traffic_type.type_name',
                          t('@traffic_type', array('@traffic_type'
                          => $liikennemuoto)), 'LIKE');

  $radiolaitteenmalli_id = $radiolaitteenmalli_id->execute();

// If-ehtolause, joka tarkistaa löytyikö tietokantataulusta käyttäjän määrit-
tämää radiomikrofonimallia. Ensimmäinen if-ehtolause, tarkistaa, onko radio-
mikrofonimalli_id-muuttuja asetettu ja onko muuttujaan asetettu olio.
  if(!empty($radiolaitteenmalli_id) &&
     $radiolaitteenmalli_id != null && is_object($radiolaitteenmalli_id)) {

// If-ehtolause, joka varmistaa, että tietokantataulusta on löytynyt malli.
    if($radiolaitteenmalli_id->rowCount() > 0) {

      $radiolaitteenmalli_id->fetchObject()->rm_model_id;

    } else {

// Asettaa radiolaitteenmalli_id-muuttujan arvoksi null, jos radiomikrofoni-
mallia ei löytynyt.
      $radiolaitteenmalli_id = null;

    } // If-ehtolause päättyy.

  } else {

// Asettaa radiolaitteenmalli_id-muuttujan arvoksi null, jos radiomikrofoni-
mallia ei löytynyt.
    $radiolaitteenmalli_id = null;

  } // If-ehtolause päättyy.

// If-ehtolause, joka tarkistaa löytyikö tietokantataulusta käyttäjän määrit-
tämää radiomikrofonimallia. Jos radiomikrofonimallia ei löytynyt tietokanta-
taulusta, tämän if-ehtolauseen sisällä lisätään tietokantatauluun käyttäjän
määrittämä radiomikrofonimalli.
  if($radiolaitteenmalli_id === '' || $radiolaitteenmalli_id == null) {
// Etsii radiomikrofonityyppi-tietokantataulusta tyypin pääavaimen id:n.
    $rm_type = db_select('ramireg_radiomicrophone_type', 'r')
    ->fields('r', array('type_id'))
    ->condition('type_name', strtolower($laitetyyppi), 'LIKE')
    ->execute()
    ->fetchObject()->type_id;
```

```
// Etsii liikennetyyppi-tietokantataulusta liikennetyypin pääavaimen id:n.
    $traffic_type = db_select('ramireg_traffic_type', 'r')
    ->fields('r', array('traffic_id'))

    ->condition('type_name', t('@traffic_type',
              array('@traffic_type' => $liikennemuoto)), 'LIKE')

    ->execute()
    ->fetchObject()->traffic_id;

// Asettaa radiomikrofonimalli-tietokantatauluun uuden radiomikrofonimallin.
    $radiolaitteenmalli_id = db_insert('ramireg_radiomicrophone_model')
    ->fields(array(
      'manufacturer' => $radiolaitteenvalmistaja,
      'model_name' => $radiolaitteenmalli,
      'amplifier_db' => $antennivahvistus,

// Asettaa käyttäjän määrittämän kanavan leveyden, jos määritelmää ei löydy,
niin syötetään tyhjä arvo. Tyhjä arvo, estää tietokantatauluun lisäämisen
uuden rivin.
      'channel_width' => (!empty($kanavanleveys) ? $kanavanleveys :
                          (!empty($kanavanleveys2) ? $kanavanleveys2 : null)),

// Asettaa käyttäjän määrittämän lähetyksen tehon, jos määritelmää ei löydy,
niin syötetään tyhjä arvo. Tyhjä arvo, estää tietokantatauluun lisäämisen
uuden rivin.
      'transmitter_power' => (!empty($lahetysteho) ? $lahetysteho :
                              (!empty($lahetysteho2) ? $lahetysteho2 : null)),

// Asettaa käyttäjän määrittämän radiomikrofonityypin, jos määritelmää ei
löydy, niin syötetään tyhjä arvo. Tyhjä arvo, estää tietokantatauluun lisää-
misen uuden rivin.
      'rm_type' => !empty($rm_type) ? $rm_type : null,

// Asettaa käyttäjän määrittämän liikennemuodon, jos määritelmää ei löydy,
niin syötetään tyhjä arvo. Tyhjä arvo, estää tietokantatauluun lisäämisen
uuden rivin.
      'traffic_type' => !empty($traffic_type) ? $traffic_type : null,
    ))
    ->execute();

  } // If-ehtolause päättyy.
```

```
// Lisää tietokantaan rekisteröityjen radiomikrofonilaitteiden tauluun uuden
laitteen. Rm_id-muuttujaan palautuu radiomikrofonilaitteen id-tunnus tieto-
kantataulussa.
  $rm_id = db_insert('ramireg_registered_radiomicrophone')
  ->fields(array(
    'uid' => $user->uid,
    'organization_id' => 1,
    'organization_access' => 1,
    'rm_group' => 1,
    'rm_model' => $radiolaitteenmalli_id,
    'rm_purpose' => $kayttotarkoitus,
    'rm_type_mark' => $tyyppimerkinta,
    'register_timestamp' => strtotime('now'),
  ))
  ->execute();

// Lisää vastaanottimien tietokantatauluun radiomikrofonin vastaanottimen
tiedot.
  $receiver_id = db_insert('ramireg_receiver')
  ->fields(array(
    'rm_id' => $rm_id,
    'uid' => $user->uid,
    'amplifier_db' => $antennivahvistus,
    'channel_width' => $kanavanleveys,
    'organization_id' => 1,
    'organization_access' => 1,
  ))
  ->execute();

// Tietokantakysely, joka hakee radiomikrofonilaitteiden sijoituspaikkatieto-
kantaulusta sijoituspaikan pääavaimen id-tunnuksen locationType-muuttujaan.
  $locationType = db_select('ramireg_location_type', 'r')
  ->fields('r', array('location_type_id'))
  ->condition('locationtype_name', t('@location_type', array('@location_type'
              => ($sijoituspaikka !== t('else, what?')
              ? $sijoituspaikka : (!empty($sijoituspaikka2)
              ? $sijoituspaikka2 : null)))), 'LIKE')
  ->execute()
  ->fetchObject()->location_type_id;

// Tietokantakysely, joka hakee postinumeroiden tietokantataulussa postinume-
ron pääavaimen id-tunnuksen.
  $postal_id = db_select('postal_codes', 'p')
  ->fields('p', array('postal_id'))
  ->condition('postal_code', $kayttopaikka, 'LIKE')
  ->execute()
  ->fetchObject()->postal_id;
```

```
// Lisää radiomikrofonilaitteiden lupahakemusten tietokantatauluun uuden lu-
pahakemuksen. Lisence_id-muuttujaan palautuu lupahakemuksen id-tunnus tieto-
kantataulussa.
  $lisence_id = db_insert('ramireg_radiomicrophone_license')
  ->fields(array(
    'uid' => $user->uid,
    'organization_id' => 1,
    'organization_access' => 1,
    'receiver_id' => $receiver_id,
    'rm_id' => $rm_id,
    'rm_purpose' => $kayttotarkoitus,
    'location_type' => $locationType,
    'postal_code' => $postal_id,
    'station_name' => $asematunnus,
    'register_timestamp' => strtotime('now'),
    'license_end' => strtotime('now') + 31536000,
  ))
  ->execute();

// Lisää radiomikrofonilaitteen aseman sijaintitauluun paikkatiedon, taajuu-
den aloitusajan sekä lopettamisajan.
  $stationLocation_id = db_insert('ramireg_station_location')
  ->fields(array(
    'start_time' => 0,
    'end_time' => 0,
    'address' => null,
    'geo_longitude' => null,
    'geo_latitude' => null,
    'postal_id' => $postal_id,
    'license_id' => $lisence_id,
  ))
  ->execute();

// Lisää taajuuksien seurantatietokantatauluun taajuuden, johon liitetään
aseman sijaintipaikkatietokantataulusta tiedot viiteavaimena ja taajuuden
vastaanottimen viiteavain.
  $frequency_id = db_insert('ramireg_transmit_frequency')
  ->fields(array(
    'transmit_frequency' => $lahetystaajuus,
    'receiver_id' => $receiver_id,
    'location_id' => $stationLocation_id,
  ))
  ->execute();

// Käyttäjä ohjautuu radiomikrofonien hallintasivulle.
  drupal_goto('rm_manager');


}          //          Lomakkeen          käsittely-funktio          päättyy.
```

Viestintäviraston radiomikrofonirekisteröintilomake

Viestintävirasto
Kommunikationsverket
Finnish Communications
Regulatory Authority

LIIKKUVAN VHF/UHF-RADIO-LAITTEEN LUPAHAKEMUS **LR**

Asiakastiedot	Hakijan nimi (sukunimi ja etunimet, toiminimi tms.)		Puhelin kotiin ( )
	Jakeluosoite		Puhelin toimeen ( )
	Postinumero ja postitoimipaikka	Y-tunnus -	Henkilötunnus -
	Asiaa hoitavan henkilön nimi		Puhelin ( )
	Lisätietoja		
Täyttöohjeita	"Henkilötunnus" täytetään vain yksityishenkilön ollessa hakijana. "Lisätietoina" tulee tarvittaessa lisäksi ilmoittaa mm. selvitys vasta-asemasta sekä mahdollinen edellinen omistaja. Tässä kohdassa esitettävä nimitieto on selvennettävä siten, että siitä ilmenee yksikäsitteisesti, onko kysymys käyttäjästä, laskutettavasta vaiko edellisestä omistajasta. "Lähetys- ja vastaanottotaajuudet" tulee ilmoittaa pareittain muodossa lähetystaajuus/vastaanottotaajuus. Tarvittaessa käytetään kohtaa "Lisätietoja".		
Liikkuvaa asemaa koskevat tiedot	Luvan ☐ ensi-hakemus ☐ uusinta ☐ muutos	Edell. luvan nro ja antovuosi	Radioverkkosuunnitelman nro
	Radiolaitteen käyttötarkoitus		
	☐ radiopuhelin ☐ radiomikrofoni ☐ radio-ohjauslaite ☐ tele-metrialaite ☐ muu, mikä		
	Radiolaitteen sijoituspaikka ☐ auto ☐ vene ☐ kannettava ☐ muu, selvitys		
	Käyttöpaikkakunta	Ehdotus aseman tunnukseksi	Antennin vahvistus, dB
	Radiolaitteen tyyppimerkintä	Lähettimien lukumäärä	Lähettimen lähtöteho, W
	Liikennemuoto ☐ 1-taaj. simpleksi ☐ 2-taaj. simpleksi ☐ dupleksi	Kanavan leveys ☐ 25 kHz ☐ 12,5 kHz ☐ muu	
	Lähetys- ja vastaanottotaajuudet		
	Lisätietoja		
Hakijan allekirjoitus	Hakija sitoutuu noudattamaan lupaehtoja ja käyttämään radiolähettimiä, jotka ovat niitä koskevien vaatimusten mukaisia. Paikka ja aika	Allekirjoitus ja nimen selvennys	
Viestintä-viraston merkinnät			

Viestintävirasto
Kommunikationsverket
Finnish Communications
Regulatory Authority
www.ficora.fi
radiotaajuudet@ficora.fi

PL 313
00181 Helsinki
Itämerenkatu 3 A
Helsinki
Puhelin (09) 69 661
Faksi (09) 6966 410
Y-tunnus 0709019-2

PB 313, FIN-00181
Helsingfors, Finland
Östersjögatan 3 A
Helsingfors, Finland
Telefon +358 9 69 661
Telefax +358 9 6966 410
FO-nummer 0709019-2

P.O. Box 313, FIN-00181
Helsinki, Finland
Itämerenkatu 3 A
Helsinki, Finland
Telephone +358 9 69 661
Telefax +358 9 6966 410
Business ID 0709019-2